TUGAS AKHIR – KI141502

# IMPLEMENTASI NEAREST NEIGHBOR PADA DATA KATEGORIK DENGAN PEMBOBOTAN ATRIBUT MENGGUNAKAN WEIGHTED SIMPLE MATCHING COEFFICIENT

ROMARIO WIJAYA
5113100062

Dosen Pembimbing I
Dr.Eng. Nanik Suciati, S.Kom.,M.Kom.

Dosen Pembimbing II
Wijayanti Nurul Khotimah, S.Kom.,M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya, 2017

**TUGAS AKHIR – KI141502**

# IMPLEMENTASI NEAREST NEIGHBOR PADA DATA KATEGORIK DENGAN PEMBOBOTAN ATRIBUT MENGGUNAKAN WEIGHTED SIMPLE MATCHING COEFFICIENT

ROMARIO WIJAYA
5113100062

Dosen Pembimbing I
Dr.Eng. Nanik Suciati, S.Kom.,M.Kom.

Dosen Pembimbing II
Wijayanti Nurul Khotimah, S.Kom.,M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

**[*Halaman ini sengaja dikosongkan*]**

**FINAL PROJECT - KI141502**

# IMPLEMENTATION OF NEAREST NEIGHBOR ON CATEGORICAL DATA WITH ATTRIBUTE WEIGHTING USING WEIGHTED SIMPLE MATCHING COEFFICIENT

ROMARIO WIJAYA
5113100062

Supervisor I
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Supervisor II
Wijayanti Nurul Khotimah, S.Kom., M.Sc.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
Sepuluh Nopember Institute of Technology
Surabaya, 2017

**[*Halaman ini sengaja dikosongkan*]**

# LEMBAR PENGESAHAN

## IMPLEMENTASI NEAREST NEIGHBOR PADA DATA KATEGORIK DENGAN PEMBOBOTAN ATRIBUT MENGGUNAKAN WEIGHTED SIMPLE MATCHING COEFFICIENT

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Komputasi Cerdas dan Visi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
**ROMARIO WIJAYA**
**NRP: 5113 100 062**

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
NIP. 197104281994122001 ..... (Pembimbing 1)

Wijayanti Nurul Khotimah, S.Kom., M.Sc.
NIP. 198603122012122004 ..... (Pembimbing 2)

**SURABAYA**
**JUNI, 2017**

**[*Halaman ini sengaja dikosongkan*]**

**IMPLEMENTASI NEAREST NEIGHBOR PADA DATA KATEGORIK DENGAN PEMBOBOTAN ATRIBUT MENGGUNAKAN WEIGHTED SIMPLE MATCHING COEFFICIENT**


| | | |
|---|---|---|
| Nama Mahasiswa | : | Romario Wijaya |
| NRP | : | 5113 100 062 |
| Jurusan | : | Teknik Informatika, FTIf ITS |
| Dosen Pembimbing 1 | : | Dr.Eng. Nanik Suciati, S.Kom., M.Kom. |
| Dosen Pembimbing 2 | : | Wijayanti Nurul Khotimah, S.Kom., M.Sc. |


## *Abstrak*


*Pada era saat ini, aplikasi komputer atau perangkat bergerak yang diperuntukkan untuk para pakar maupun yang diperuntukkan untuk orang awam terus dikembangkan ke arah yang lebih cerdas untuk memudahkan pekerjaan manusia. Aplikasi yang dibuat haruslah memiliki kemampuan untuk memberikan rekomendasi hasil analisa input. Contoh input yang paling sering dianalisa adalah data, baik itu data numerik maupun data kategorik. Di sinilah peran klasifikasi sangat dibutuhkan oleh para pembuat aplikasi untuk terus mengembangkan aplikasinya agar semakin cerdas dan diminati banyak orang.*

*Metode klasifikasi yang paling umum, simpel dan mudah untuk diterapkan salah satunya adalah k-Nearest-Neighbor (k-NN). Algoritma k-NN dinilai cepat karena merupakan Instance-based learning atau lazy-learning yang tidak memerlukan fase latihan yang panjang pada awal proses. Namun dalam penerapannya, untuk melakukan klasifikasi terhadap data kategorik, diperlukan cara khusus dalam menghitung jarak antar data yang ada. Salah satu cara khusus dalam menghitung jarak antar data yang paling sering diterapkan adalah menggunakan Simple Matching Coefficient (SMC). Dalam penelitian kali ini, akan diimplementasikan sebuah metode klasifikasi terhadap data*

*kategorik yang menggunakan metode klasifikasi Nearest-Neighbor dengan pembobotan atribut menggunakan Weighted Simple Matching Coefficient (WSMC). Dengan diimplementasikannya metode kali ini, diharapkan dapat mendapatkan hasil klasifikasi yang optimal.*

*Uji coba yang dilakukan terhadap 6 dataset dengan atribut bersifat kategorik, menunjukkan kemampuan metode dalam melakukan klasifikasi. Dengan rata-rata akurasi tertinggi didominasi oleh metode lokal. Rata-rata akurasi tertinggi untuk dataset Nursery, Cars, Gerakan tangan, Soybeans, Vote dan Dhermatology berturut-turut sebesar 77.31%, 80.35%, 97.00%, 91.64%, 92.41%, dan 95.90%.*

***Kata kunci: Nearest-Neighbor, data kategorik, pembobotan atribut.***

**IMPLEMENTATION OF NEAREST NEIGHBOR ON CATEGORICAL DATA WITH ATTRIBUTE WEIGHTING USING WEIGHTED SIMPLE MATCHING COEFFICIENT**


| | |
|---|---|
| Student Name | : Romario Wijaya |
| Registration Number | : 5113 100 062 |
| Department | : Informatics Engineering, FTIf ITS |
| First Supervisor | : Dr.Eng. Nanik Suciati, S.Kom.,M.Kom. |
| Second Supervisor | : Wijayanti Nurul Khotimah, S.Kom., M.Sc. |


## *Abstract*


*In this day, computer application or mobile application for the expert or for the common people is always developed in many smart ways to perform such a big help for people. The application that was made must have an ability to perform such input analyic recommendation. One of input type is data, the type can be either numeric or categorical. And this time, the use of classification is needed to make the application more smart and demanded by many people.*

*One of classification method that very common, and simple to be implemented is k-Nearest Neighbor(k-NN). K-NN Algorithm is considered fast enough because it is Instance based learning or lazy-learning classification which doesn't need to perform such long training phase in the beginning process.But, in the further implementation for categorical data, a specific method is needed to calculate distance between the data. One of the method that is commonly used is Simple Matching Coefficient (SMC). In this research, a method of classification against categorical data using k-NN with attribute weighting using Weighted Simple Matching Coefficient (WSMC) method is implemented. By the implementation of this method, hopefully we can get the optimal classification result.*

*The testing phase is done against 3 set of data with attribute construct in categorical value. It's showing that the abillity of this*

*method to perform classification. With the highest average of accuracy is dominated by local method. The highest average of accuracy for set of data Nursery, Cars, Hand Movement, Soybeans, Vote, and Dhermatology are 77.31%, 80.35%, 97.00%, 91.64%, 92.41%, and 95.90% respectively.*

***Keywords: Nearest-Neighbor, categorical data, attribute weighting.***

# KATA PENGANTAR

Puji syukur kepada Tuhan yang Maha Esa, atas berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul **"Implementasi Nearest Neighbor pada Data Kategorik dengan Pembobotan Atribut Menggunakan Weighted Simple Matching Coefficient"**.

Buku tugas akhir ini disusun dengan harapan dapat memberikan manfaat dalam proses klasifikasi data kategorik khususnya untuk data fitur gerakan tangan guna penerjemahan bahasa isyarat. Selain itu, penulis berharap dapat memberikan kontribusi positif bagi kampus Teknik Informatika ITS.

Dalam perancangan, pengerjaan, dan penyusunan tugas akhir ini, penulis banyak mendapatkan bantuan dari berbagai pihak. Penulis ingin mengucapkan terima kasih kepada:

1. Dr.Eng. Nanik Suciati, S.Kom., M.Kom. dan Wijayanti Nurul Khotimah, S.Kom., M.Sc. selaku dosen pembimbing penulis yang telah memberi ide, nasihat dan arahan sehingga penulis dapat menyelesaikan tugas akhir dengan tepat waktu.
2. Orang tua penulis yang telah memberikan dukungan moral, spiritual dan material serta senantiasa memberikan doa demi kelancaran dan kemudahan penulis dalam mengerjakan tugas akhir.
3. Saudara kandung: kakak dan keluarga serta seluruh keluarga besar yang telah memberikan dukungan yang besar baik secara langsung maupun secara implisit.
4. Teman-teman sepermainan: teman-teman yang juga menjalani mata kuliah Tugas Akhir yang sering bermain dan berkumpul bersama dan saling menguatkan.
5. Teman satu bimbingan Bu Nanik dan Bu Wijayanti yang telah memberikan dukungan berupa semangat maupun kritik dan saran.

6. Teman teman dari GKT Filadelfia Mojokerto dan GKT Anugerah Surabaya yang selalu memberikan dukungan baik doa maupun kekuatan.
7. Pihak-pihak lain yang tidak bisa penulis sebutkan satu-persatu.

Penulis menyadari masih ada kekurangan dalam penyusunan tugas akhir ini. Penulis mohon maaf atas kesalahan, kelalaian maupun kekurangan dalam penyusunan tugas akhir ini. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan ke depan.

Surabaya, Juni 2017

Penulis

# DAFTAR ISI

# DAFTAR GAMBAR

**[*Halaman ini sengaja dikosongkan*]**

# DAFTAR TABEL

**[*Halaman ini sengaja dikosongkan*]**

# DAFTAR KODE SUMBER

**[*Halaman ini sengaja dikosongkan*]**

# BAB I
# PENDAHULUAN

Pada bab ini dibahas hal-hal yang mendasari tugas akhir. Bahasan meliputi latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika laporan tugas akhir.

## 1.1 Latar Belakang

Pada era saat ini, aplikasi komputer atau perangkat bergerak yang diperuntukkan untuk para pakar maupun yang diperuntukkan untuk orang awam terus dikembangkan ke arah yang lebih cerdas untuk memudahkan pekerjaan manusia. Aplikasi yang dibuat haruslah memiliki kemampuan untuk memberikan rekomendasi hasil analisa input. Contoh input yang paling sering dianalisa adalah data, baik itu data numerik maupun data kategorik. Di sinilah peran klasifikasi sangat dibutuhkan oleh para pembuat aplikasi untuk terus mengembangkan aplikasinya agar semakin cerdas dan diminati banyak orang.

Metode klasifikasi yang paling umum, simpel dan mudah untuk diterapkan salah satunya adalah k-Nearest-Neighbor (k-NN). Algoritma k-NN dinilai cepat karena merupakan *Instance-based learning* atau *lazy-learning* [1] yang tidak memerlukan fase latihan pada awal proses [2]. Namun dalam penerapannya, untuk melakukan klasifikasi terhadap data kategorik, diperlukan cara khusus dalam menghitung jarak antar data yang ada. Salah satu cara khusus dalam menghitung jarak antar data yang paling sering diterapkan adalah menggunakan Simple Matching Coefficient (SMC)[3].

Dalam penelitian kali ini, akan diimplementasikan sebuah metode klasifikasi terhadap data kategorik yang menggunakan metode klasifikasi Nearest-Neighbor dengan pembobotan atribut menggunakan Weighted Simple Matching Coefficient (WSMC). Dengan diimplementasikannya metode kali ini, diharapkan dapat mendapatkan hasil klasifikasi yang optimal.

## 1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana cara memberikan bobot untuk atribut yang nilainya berupa data kategorik menggunakan WSMC?
2. Bagaimana cara melakukan klasifikasi terhadap data kategorik menggunakan k-NN dengan pembobotan atribut menggunakan WSMC?
3. Bagaimana performa metode klasifikasi k-NN dengan pembobotan atribut menggunakan WSMC untuk klasifikasi data kategorik?

## 1.3 Batasan Masalah

1. Implementasi kode akan menggunakan IDE Matlab.
2. Dataset yang digunakan adalah dataset hasil ekstraksi fitur gerakan tangan menggunakan Kinect 2.0, dataset *nursery*, dataset *Dhermatology*, dataset *Vote*, dataset *Soybeans* dan dataset *cars*.
3. Metode klasifikasi yang digunakan adalah k-NN dengan pembobotan atribut menggunakan WSMC.
4. Alat bantu yang akan dipakai untuk hasil uji perbandingan adalah WEKA.
5. Dataset yang digunakan bersifat data kategorik.
6. Dataset fitur dinamis gerakan tangan yang dipakai adalah dataset dari Tugas Akhir Yahya Eka Nugyasa.
7. Proses Klasifikasi untuk dataset Gerakan Tangan hanya sampai penentuan jenis gerakan tangan.

## 1.4 Tujuan Tugas Akhir

Tujuan dari pembuatan Tugas Akhir ini adalah melakukan implementasi dan uji coba metode klasifikasi k-NN dengan pembobotan atribut menggunakan WSMC.

## 1.5 Manfaat Tugas Akhir

Tugas akhir ini diharapkan dapat membuktikan efektivitas metode k-NN dengan pembobotan atribut menggunakan WSMC untuk mengklasifikasikan data yang bersifat kategorik.

## 1.6 Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. Studi Literatur

   Pada tahap ini, akan dicari literatur yang relevan untuk dijadikan referensi dalam pengerjaan tugas akhir. Studi literatur dapat diambil dari buku, internet, ataupun materi dalam suatu mata kuliah yang berhubungan dengan metode yang akan digunakan.

2. Analisis dan Desain Perangkat Lunak

   Proses dimulai dari data masukan berupa dataset yang bersifat kategorik. Dilanjutkan dengan pemberian bobot kepada tiap atribut menggunakan 4 persamaan yang ditentukan oleh *Global Entropy (GE)*, *Global Gini Diversity (GG)* dan *Local Entropy (LE)*, *Local Gini Diversity (LG)*. Atribut yang telah diberi bobot akan diklasifikasikan menggunakan algoritma Nearest-Neighbor. Keluaran dari program berupa kelas dari masing-masing data yang telah di masukkan.

3. Implementasi Perangkat Lunak

   Pembuatan program akan dilakukan dengan menggunakan bahasa pemrograman Matlab, Integrated Development Environment (IDE) Matlab IDE.

4. Uji Coba dan Evaluasi

   Pengujian dan evaluasi akan menggunakan alat bantu Weka untuk membandingkan dengan metode klasifikasi yang lain dengan memberikan data uji coba yang sama.

## 1.7 Sistematika Laporan

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut:

**Bab I Pendahuluan**

Bab yang berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

**Bab II Dasar Teori**

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

**Bab III Analisis dan Perancangan**

Bab ini berisi tentang analisis dan perancangan desain sistem klasifikasi Nearest-Neighbour dengan pembobotan atribut untuk data kategorik.

**Bab IV Implementasi**

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode yang digunakan untuk proses implementasi.

**Bab V Uji Coba dan Evaluasi**

Bab ini membahas tahap-tahap uji coba. Kemudian hasil uji coba dievaluasi untuk kinerja dari aplikasi yang dibangun.

**Bab VI Kesimpulan dan Saran**

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan aplikasi ke depannya.

# BAB II
# DASAR TEORI

## 2.1 Data Kategorik

Data kategorik adalah data yang memiliki dua atau lebih nilai kategori, namun tidak memiliki urutan nilai yang intrinsik. Sebagai contoh jenis kelamin dan warna rambut termasuk jenis data kategorik, karena atribut tersebut memiliki banyak nilai (Laki-laki atau Perempuan untuk jenis kelamin. Hitam, Coklat, Pirang, dan sebagainya untuk warna rambut), namun tidak memiliki urutan yang jelas tentang yang mana yang memiliki nilai lebih besar atau lebih kecil, atau nilai mana yang lebih dekat dengan nilai yang lain. Sebuah data kategorik yang memiliki urutan, adalah disebut dengan data ordinal[4]. Tabel contoh data kategorik dapat dilihat pada **Tabel 2.1.**

**Tabel 2.1** Contoh Data Kategorik

| Atribut #1 | Atribut #2 | Atribut #3 | Atribut #4 | Atribut #5 | Atribut #6 | Label Kelas |
|---|---|---|---|---|---|---|
| Vhigh | Vhigh | 2 | 2 | Small | Low | unacc |
| Vhigh | Vhigh | 2 | 2 | Small | Med | unacc |
| Vhigh | Vhigh | 2 | 2 | Small | High | unacc |

Pada **Tabel 2.1** atribut #3 dan atribut #4 terlihat berbentuk numerik, namun angka di sana hanyalah menunjukkan nomor kategori saja, dan tidak perlu dihitung menggunakan operasi matematika seperti tambah (+), kurang (-), kali (*), atau bagi (/).

## 2.2 Fitur Dinamis Gerakan Tangan

Bahasa isyarat adalah sarana berkomunikasi bagi penderita tuna rungu. Bahasa isyarat dikembangkan dan memiliki karakteristik sendiri di berbagai negara. Contoh Bahasa Isyarat dapat dilihat pada **Gambar 2.1**.


**Gambar 2.1** Contoh bahasa isyarat

Dalam gerakan pada **Gambar 2.1** terdapat bermacam-macam bentuk gerakan tangan seperti melingkar, rotasi, persegi dan lain sebagainya. Bentuk-bentuk gerakan tangan tersebut disebut dengan fitur dinamis gerakan tangan. Dalam tugas akhir ini, fitur dinamis gerakan tangan didapatkan dari ekstraksi fitur yang telah dilakukan oleh Yahya Eka Nugyasa pada penelitian sebelumnya. Fitur

gerakan tangan tersebut dapat dimanfaatkan untuk mendapatkan hasil berupa gerakan melingkar, rotasi, dan sebagainya. Contoh hasil ekstraksi fitur gerakan tangan dapat dilihat pada **Tabel 2.2**.

**Tabel 2.2** Contoh Data Hasil Ekstraksi Fitur Gerakan Tangan

| F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | F10 |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 2 | 2 | 2 | 8 | 6 | 6 | 4 | 7 | 7 |
| 2 | 2 | 8 | 8 | 6 | 6 | 4 | 4 | 1 | 1 |

Pada **Tabel 2.2** F1-F10 menunjukkan nomor frame. Data berupa angka yang mengodekan suatu kategori gerakan dengan ketentuan sebagai berikut :

- Angka 1 menunjukkan gerakan ke arah serong kanan atas.
- Angka 2 menunjukkan gerakan ke arah atas.
- Angka 3 menunjukkan gerakan ke arah serong kiri atas.
- Angka 4 menunjukkan gerakan ke arah kiri.
- Angka 5 menunjukkan gerakan ke arah serong kiri bawah.
- Angka 6 menunjukkan gerakan ke arah bawah.
- Angka 7 menunjukkan gerakan ke arah serong kanan bawah.
- Angka 8 menunjukkan gerakan ke arah kanan[5].

## 2.3 Simple Matching Coefficient (SMC)

Simple Matching Coefficient (SMC) adalah perhitungan statistik yang digunakan untuk membandingkan kesamaan dan perbedaan sampel set. Perhitungan ini sangat penting untuk melakukan klasifikasi terhadap data yang memiliki atribut non-numerik seperti teks atau kategori golongan.

Sebagai contoh jika diberikan dua objek *A* dan *B*, masing-masing dengan *n* atribut biner, SMC dapat dituliskan pada persamaan **(2.1)**.

$$SMC = \frac{Jumlah\ Atribut\ yang\ tidak\ sama}{Jumlah\ Atribut} = \frac{M_{01} + M_{10}}{M_{00} + M_{01} + M_{10} + M_{11}} \quad (2.1)$$

$M_{11}$ merepresentasikan jumlah atribut dimana atribut pada objek *A* dan *B* memiliki nilai 1. $M_{01}$ merepresentasikan jumlah atribut dimana atribut pada objek *A* memiiki nilai 0 dan atribut pada objek *B* memiliki nilai 1. $M_{10}$ merepresentasikan jumlah atribut dimana atribut pada objek *A* memiliki nilai 1 dan atribut pada objek *B* memiliki nilai 0. $M_{00}$ merepresentasikan jumlah atribut dimana atribut pada objek *A* dan *B* memiliki nilai 0[6].

## 2.4 Weighted Simple Matching Coefficient (WSMC)

Weighted Simple Matching Coefficient (WSMC) adalah perhitungan statistik yang hampir sama dengan Simple Matching Coefficient namun dalam implementasinya, setiap atribut tidak akan dihitung secara rata, namun dengan memberikan bobot terhadap tiap atribut sehingga secara tidak langsung mengurangi pengaruh data yang akan mengacaukan klasifikasi. Metode perhitungan ini dapat digunakan secara efektif untuk melakukan klasifikasi terhadap data yang bersifat kategorik[3]. Pengukuran jarak baru akan dinotasikan dengan $WSMC_{global}$ dan $WSMC_{local}$ tergantung dari metode pembobotan(global atau lokal) yang digunakan.

Pada metode global, atribut dikaitkan dengan vektor bobot $\varpi = (\omega_1, \ldots, \omega_d, \ldots, \omega_D)$ dan jarak WSMC antara objek data $x_i$ dan $x_j$ diberikan pada persamaan **(2.2)**.

$$WSMC_{global}(x_i, x_j, \varpi) = \sum_{d=1}^{D} \omega_d \times I(x_{id} \neq x_{jd}) \quad (2.2)$$

Dimana *I*(.) adalah fungsi indikator dengan *I*(*true*) = 1 dan *I*(*false*) = 0, dan $\omega_d$ adalah bobot unik yang diberikan kepada atribut *d* yang akan dihitung berdasarkan dari *Global Entropy* (*GE*) dan *Global Gini diversity index* (*GG*). Dari perspektif peringkat fitur, nilai *Entropy* dan *Gini diversity index* di sini mengukur derajat kontribusi dari atribut dalam mendiskriminasikan kelas *training*. Secara formal, *Entropy* dan *Gini diversity index* dapat dihitung dengan persamaan **(2.4)** dan **(2.6)**.

Pada metode lokal, tujuannya adalah memberikan bobot yang didapat berdasarkan kelas untuk tiap atribut, yang mengindikasikan kontribusi yang berbeda oleh tiap atribut terhadap kelas yang berbeda. Pada kasus ini, kelas ke-*m* dikaitkan dengan vektor bobot $\boldsymbol{w}_m = (\boldsymbol{w}_{m1}, \dots, \boldsymbol{w}_{md}, \dots, \boldsymbol{w}_{mD})$ dengan $0 \leq \boldsymbol{w}_{md} \leq 1$ untuk $d = 1,2,...D$. jarak SMC antara objek data $\boldsymbol{x}_i$ dan $\boldsymbol{x}_j$ berubah menurut persamaan

**(2.3)**.

$$WSMC_{local}(x_i, x_j, w_m) = \sum_{d=1}^{D} w_{md} \times I(x_{id} \neq x_{jd}) \qquad \textbf{(2.3)}$$

Bobot lokal dapat dihitung juga menggunakan informasi *Entropy* dan *Gini diversity index*, seperti pada bobot global, namun dengan menggunakan *Local Entropy* (*LE*) dan *Local Gini diversity index* (*LG*). Adapun *LE* dan *LG* didefinisikan dalam persamaan **(2.10)** dan **(2.12)**

$$GE(s_d) = -\sum_{m=1}^{M} p(m|s_d) \log_2 p(m|s_d) \qquad \textbf{(2.4)}$$

Dengan $s_d$ menotasikan sebuah kategori ke-*d*, M adalah jumlah kelas yang ada pada dataset dan $p(m/s_d)$ didefinisikan dengan persamaan (2.5).

$$p(m/s_d) = \frac{\sum_{(x,y)\in c_m} I(x_d = s_d)}{\sum_{(x,y)\in tr} I(x_d = s_d)} \quad \textbf{(2.5)}$$

Pada persamaan **(2.5)** dilakukan perhitungan terhadap tiap data yang pada kelas *m* yang memiliki atribut ke *d* bernilai $s_d$ dan dibagi dengan jumlah semua data yang memiliki atribut ke *d* bernilai $s_d$. Persamaan **(2.4)** disebut *Global Entropy* karena dihitung menggunakan $p(m/s_d)$ yang menyebabkan nilainya unik terhadap tiap kelas yang ada *m*. Hal ini menyebabkan persamaan **(2.6)** juga disebut sebagai *Global Gini diversity index* mengingat perhitungan juga menggunakan $p(m/s_d)$.

$$GG(s_d) = 1 - \sum_{m=0}^{M} [p(m|s_d)]^2 \quad \textbf{(2.6)}$$

Dengan $p(m|s_d)$ dihitung dengan persamaan (2.5).

Perhitungan persamaan **(2.4)** dan **(2.6)** berpengaruh besar terhadap perhitungan *Global Weight* ($\omega_d$). Apabila $\omega_d$ dihitung menggunakan *GE* akan menghasilkan $\omega_d^{(GE)}$ yang dapat dihitung menggunakan persamaan **(2.7)**, sedangkan apabila $\omega_d$ dihitung menggunakan *GG* akan menghasilkan $\omega_d^{(GG)}$ yang dapat dihitung menggunakan persamaan **(2.9)**.

Pada persamaan **(2.7)** dan **(2.9)**, muncul sebuah variabel baru $0 \leq p(s_d) \leq 1$ yang merupakan derajat pengaruh atribut $s_d$ terhadap semua data yang digunakan. Adapun $p(s_d)$ dapat dihitung menggunakan persamaan **(2.8)**.

$$\omega_d^{(GE)} = e^{\frac{1}{\log_2 M} \sum_{s_d \in S_d} p(s_d) \times GE(s_d)} \quad \textbf{(2.7)}$$

$$p(s_d) = \frac{1}{N} \sum_{(x,y)\in tr} I(x_d = s_d) \quad \textbf{(2.8)}$$

Dengan ***tr*** adalah *training dataset* dan ***N*** menunjukkan jumlah data.

$$\omega_d^{(GG)} = e^{\frac{M}{M-1}\sum_{s_d \in S_d} p(s_d) \times GG(s_d)} \quad \textbf{(2.9)}$$

$$LE(m,d) = -\sum_{s_d \in S_d} p(s_d|m) \log_2 p(s_d|m) \quad \textbf{(2.10)}$$

Dengan $p(s_d|m)$ didefinisikan dengan persamaan **(2.11)**.

$$\boldsymbol{p}(s_d|m) = \frac{1}{|C_m|} \sum_{(x,y) \in C_m} I(x_d = s_d) \quad \textbf{(2.11)}$$

Dengan $\boldsymbol{C_m}$ merupakan kelas ke-m dan $|\boldsymbol{C_m}|$ adalah jumlah sampel pada kelas ke-m.

$$LG(m,d) = 1 - \sum_{s_d \in S_d} [p(s_d|m)]^2 \quad \textbf{(2.12)}$$

Dengan $p(s_d|m)$ dihitung menggunakan persamaan **(2.11)**.

Sama halnya dengan perhitungan pada GE dan GG, perhitungan LE dan LG pada persamaan **(2.10)** dan **(2.12)** tidak jauh berbeda. Yang memberikan perbedaan pada metode Global dan Lokal adalah pada perhitungan derajat kontribusi tiap atribut terhadap tiap kelas. Pada metode Global, digunakan $p(m/s_d)$ yang perhitungannya melibatkan semua data yang ada pada data *training*. Berbeda dengan metode Global, pada metode Lokal dihitung menggunakan $\boldsymbol{p}(s_d|m)$ yang tidak melibatkan semua data

yang ada pada data *training* namun hanya menghitung spesifik terhadap satu kelas khusus saja.

$$w_{md}^{(LE)} = e^{\frac{1}{\log_2 |S_d|} \times LE(m,d)} \quad \textbf{(2.13)}$$

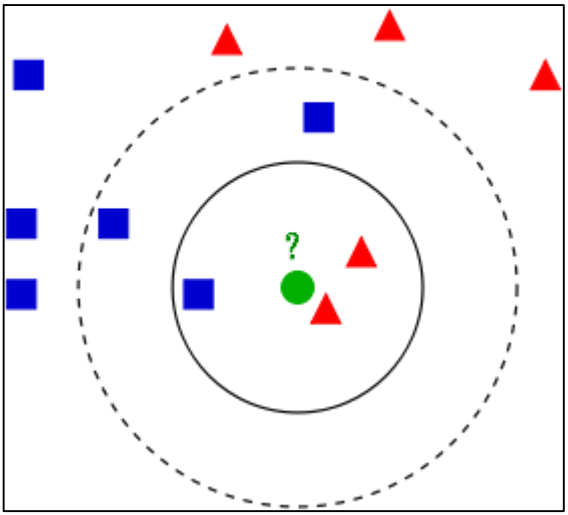
$$w_{md}^{(LG)} = e^{\frac{|S_d|}{|S_d|-1} \times LG(m,d)} \quad \textbf{(2.14)}$$

Berbeda dengan pembobotan global, bobot lokal yang masing-masing dihitung dengan persamaan **(2.13)** dan **(2.14)** tidak menggabungkan nilai dari tiap-tiap *LE* dan *LG* yang sudah dihitung seperti yang dilakukan pada metode global. Pada pembobotan lokal, bobot akan dihitung secara berulang kali terhadap tiap *LE* dan *LG*, sehingga akan didapatkan matriks bobot yang ukurannya lebih besar, namun menunjukkan hubungan antara sebuah atribut dengan masing-masing kategori label kelas secara spesifik dan unik.

## 2.5 k-Nearest Neighbor

Dalam pengenalan pola, algoritma k-Nearest Neighbors (k-NN) adalah metode non-parametrik digunakan untuk klasifikasi dan regresi. Input terdiri dari k contoh data pelatihan dengan beberapa fitur. Output tergantung pada apakah k-NN digunakan untuk klasifikasi atau regresi.

Dalam klasifikasi k-NN, output adalah keanggotaan kelas. Sebuah objek diklasifikasikan berdasarkan suara terbanyak dari tetangganya, objek akan ditandai dengan kelas yang paling umum (mayoritas) di antara k tetangga terdekatnya (k adalah bilangan bulat positif, biasanya kecil). Jika k = 1, maka objek akan langsung ditandai dengan kelas tetangga yang paling dekat. Dalam regresi k-NN, output adalah nilai properti untuk objek. Nilai ini adalah rata-rata nilai dari k tetangga terdekatnya. Ilustrasi dapat dilihat pada **Gambar 2.2**.




**Gambar 2.2** Ilustrasi k-NN

K-NN adalah berbasis *Instance Based Learning*, atau *Lazy Learning*, di mana fungsi ini hanya didekati secara lokal dan semua perhitungan ditangguhkan sampai klasifikasi. Algoritma k-NN adalah salah satu yang paling sederhana dari semua algoritma *machine learning*[1].

Data pelatihan (*data training*) terdiri dari vektor-vektor dengan fitur multidimensi yang sudah terlabel dengan properti kelas. Fase pelatihan hanya digunakan sebagai fase penyimpanan vektor data pelatihan.

Pada tahap klasifikasi, k adalah konstanta yang ditentukan sesuai dengan kebutuhan. Nilai k digunakan untuk menentukan berapa banyak tetangga terdekat yang akan diperhitungkan "suara" nya, untuk ikut ambil bagian menentukan label kelas data yang akan diklasifikasikan.

Perhitungan jarak antar tetangga (vektor) yang biasa digunakan untuk variabel kontinyu (numerik) adalah jarak Euclidean. Untuk variabel diskrit, seperti untuk klasifikasi teks, metrik lain dapat digunakan, seperti *overlap metric* (atau jarak

Hamming). Dalam konteks data ekspresi gen, misalnya, k-NN juga telah digunakan dengan koefisien korelasi seperti Pearson dan Spearman[7]. Seringkali, akurasi klasifikasi k-NN dapat ditingkatkan secara signifikan jika metrik jarak dipelajari dengan algoritma khusus.

## 2.6 k-Nearest Neighbor dengan SMC

k-Nearest Neighbor dengan SMC adalah sebuah metode pengembangan dari k-NN tradisional yang menggunakan perhitungan jarak khusus (SMC) untuk menghitung jarak antar tiap atributnya. Pengembangan ini dilakukan untuk membuat k-NN dapat bekerja pada atribut dengan nilai yang bersifat kategorik. *Pseudocode* k-Nearest Neighbor dengan SMC dapat dilihat pada **Algoritma 2.1.**

Pada **Algoritma 2.1** data masukan berupa data tes, dihitung masing-masing jaraknya terhadap data *training*. Kemudian dilakukan pengurutan sampel dari kelas data *training* dimulai dari yang jaraknya paling sedikit. Kemudian di ambil data teratas sejumlah *k*. Keluaran algoritma berupa label kelas terbanyak.

**Input :** *tr*, the number of nearest neighbors *k*, and test sample z = (x,y)
**Output :** *y*, the class label of x
**Begin**
Compute SMC (x, $x_i$), the distance between z and each sample ($x_i$,$y_i$) ∈ tr;
Select $NN_z \subseteq$ tr, k the set of *k* closest training samples to z, in terms of the distances;
Output y = $\text{argmax}_m \sum_{(x_j,y_j)\in NN_z} I(m = y_j)$
**End**

**Algoritma 2.1** k-NN dengan SMC

## 2.7 k-Nearest Neighbor dengan WSMC

k-Nearest Neighbor dengan WSMC adalah sebuah metode pengembangan dari k-NN dengan SMC, dimana di dalam k-

Nearest Neighbor dengan SMC, atribut yang ada dihitung begitu saja dengan bobot yang sama rata. Dalam pengembangan metode ini, perhitungan terhadap jarak antar atribut tetap menggunakan SMC, namun dengan tambahan pembobotan untuk tiap atributnya, sehingga tiap atribut memiliki bobot peran yang berbeda dalam menentukan hasil klasifikasi. Algoritma *learning* dapat dilihat pada **Algoritma 2.2** dan **Algoritma 2.3**.

Pada **Algoritma 2.2** bobot dari tiap atribut akan dihitung menggunakan metode global, sehingga bobot yang dipakai adalah $\omega_d^{(GE)}$ (jika pengguna memilih menggunakan fitur *Entropy*) atau $\omega_d^{(GG)}$ (jika pengguna memilih menggunakan fitur *Gini Diversity index*). Bobot $\omega_d$ yang telah digantikan oleh $\omega_d^{(GE)}$ atau $\omega_d^{(GG)}$ selanjutnya akan digunakan untuk melakukan klasifikasi dengan cara mengalikan bobot atribut dengan jarak yang sudah dihitung menggunakan SMC untuk tiap atributnya.

Sedangkan untuk **Algoritma 2.3**, bobot dari tiap atribut akan dihitung menggunakan metode lokal, sehingga bobot yang dipakai adalah $w_{md}^{(LE)}$(jika pengguna memilih menggunakan fitur *Entropy*) atau $w_{md}^{(LG)}$ (jika pengguna memilih menggunakan fitur *Gini Diversity index*). Berbeda dengan metode global, pada metode lokal bobot yang terbentuk tidak hanya satu untuk tiap atribut, namun sejumlah label kelas yang ada. Hal ini dimaksudkan agar nantinya ketika data diklasifikasikan, bobot atribut yang dipakai adalah bobot atribut yang sesuai dengan karakter label kelas tertentu. Setelah bobot lokal terbentuk, dilakukan proses klasifikasi dengan cara mengalikan jarak yang dihitung menggunakan SMC dengan bobot lokal untuk masing-masing label kelas.

**Input:** $tr = \cup_{m=1}^{M} c_m$ , and GE or GG specified for different weighting methods.
**Output:** $\varpi = < \omega_1, \ldots, \omega_d, \ldots, \omega_D >$
**begin**
  **for** d=1 **to** D **do**
    **if** GE is specified **then**
      $\omega_d \leftarrow \omega_d^{(GE)}$ using Eq. **(2.7)**
    **else**
      $\omega_d \leftarrow \omega_d^{(GG)}$ using Eq. **(2.9)**
  **end**
**end**

**Algoritma 2.2** *Global Weights Learning Algorithm* GWLA

**Input:** $c_m$ , and LE or LG specified for different weighting methods.
**Output:** $w_m = < w_{m1}, \ldots, w_{md}, \ldots, w_{mD} >$
**begin**
  **for** d=1 **to** D **do**
    **if** LE is specified **then**
      $\varpi_{md} \leftarrow w_{md}^{(LE)}$ using Eq. **(2.13)**
    **else**
      $\varpi_{md} \leftarrow w_{md}^{(LG)}$ using Eq. **(2.14)**
  **end**
  **for** d=1 **to** D **do**
    $w_{md} \leftarrow \frac{\varpi_{md}}{\sum_{d=1}^{D} \varpi_{md}}$ , such that $\|\mathbf{w}_m\|_1 = 1$.
  **end**
**end**

**Algoritma 2.3** *Local Weights Learning Algorithm* LWLA

## 2.8 Matlab

Matlab adalah bahasa dengan performa tinggi yang digunakan untuk komputasi teknis. Matlab terintegrasi dengan perhitungan, visualisasi, dan pemrograman dalam lingkungan yang mudah digunakan di mana masalah dan solusi dinyatakan dalam notasi matematika yang umum. Penggunaan Matlab meliputi :

- Matematika dan komputasi
- Pengembangan Algoritma
- Pemodelan dan simulasi
- Analisis data, eksplorasi dan visualisasi
- Rekayasa grafis
- Pembuatan aplikasi termasuk pembuatan GUI[8]

## 2.9 Weka

Weka merupakan perangkat lunak *open source* yang dikeluarkan di bawah GNU (*General Public License)* yang berisi koleksi algoritma *machine learning* untuk tugas-tugas *data mining*. Algoritma dapat diterapkan langsung ke dataset atau dipanggil dari kode Java. Weka berisi alat untuk *data pre-processing*, klasifikasi, regresi, *clustering*, aturan asosiasi, dan visualisasi. Weka juga cocok untuk mengembangkan skema *machine learning* baru[9].

## 2.10 Cross Validation

Cross Validation adalah model teknik validasi yang digunakan untuk melakukan validasi suatu metode terhadap data pengujian yang independen. Pengujian biasanya diatur untuk sebuah metode yang tujuan utamanya adalah prediksi seperti klasifikasi. Validasi dilakukan dengan tujuan untuk mengukur seberapa akurat metode prediksi yang digunakan untuk tiap dataset yang diuji cobakan.

Dalam dataset uji coba untuk klasifikasi, terdapat dua data yang dijadikan data masukan yaitu, data *training* dan data *testing*. Ide utama dari Cross Validation adalah membuat dua partisi

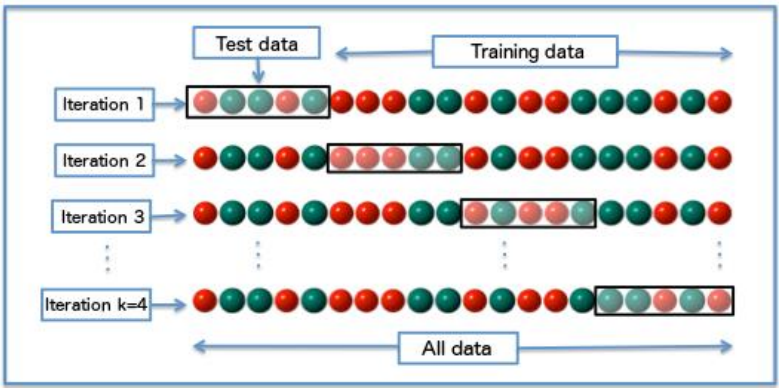
masing-masing dengan persentase tertentu (data *training* 70%, data *testing* 30% atau data *training* 60%, data *testing* 40% dan lain sebagainya). Nantinya, proses klasifikasi yang dilakukan akan menggunakan data hasil pembagian partisi yang sudah dilakukan sebelumnya.

## 2.11 K-Fold Cross Validation

K-Fold Cross Validation adalah model teknik validasi yang menggunakan metode Cross Validation sebagai dasar tekniknya. Yang membedakan di sini adalah konstanta *k* yang dipakai. Konstanta *k* menunjukkan berapa banyak partisi yang akan dibentuk dari data masukan yang diberikan. Dengan satu partisi ditentukan sebagai data *testing* dan *k*-1 partisi sisanya digunakan sebagai data *training*.

Berbeda dengan Cross Validation yang uji cobanya hanya dilakukan satu kali, pada k-Fold Cross Validation, uji coba dilakukan sebanyak *k* kali iterasi dengan menggilir tiap partisi yang sudah dibentuk mulai dari partisi pertama sebagai data *testing*, partisi lainnya menjadi data *training*. Dilanjutkan dengan partisi kedua menjadi data *testing*, partisi lainnya menjadi data *training* dan seterusnya sampai iterasi ke-*k*. Data akurasi yang diukur tiap iterasi nantinya akan dirata-rata untuk menjadi data akurasi yang dianggap sah sebagai hasil uji coba.

Ilustrasi untuk k-Fold Cross Validation dapat dilihat pada **Gambar 2.3**. Dapat dilihat contoh dari ilustrasi menggunakan data dengan jumlah data sebanyak 20 data. Nilai *k* ditetapkan adalah 4. Pada iterasi pertama, data dibagi menjadi 4 bagian, yang mana adalah nilai *k*. Dengan bagian pertama menjadi data *testing* dan bagian sisanya menjadi data *training*. Untuk iterasi kedua digunakan bagian kedua sebagai data *testing* dan sisanya menjadi data *training* dan seterusnya sampai iterasi keempat[10].




**Gambar 2.3** Ilustrasi k-Fold Cross Validation

**[*Halaman ini sengaja dikosongkan*]**

# BAB III
# ANALISIS DAN PERANCANGAN

Bab analisis dan perancangan berisi analisis kebutuhan dan perancangan aplikasi yang akan dibangun. Tahap analisis membahas mengenai analisis kebutuhan yang menjadi dasar dari tahap perancangan.

## 3.1 Tahap Analisis

Tahap analisis mendefinisikan kebutuhan yang akan dipenuhi dalam implementasi metode klasifikasi k-NN dengan pembobotan atribut untuk data kategorik.

### 3.1.1 Deskripsi Umum

Pada tugas akhir ini, akan diimplementasikan sebuah metode klasifikasi k-NN yang menggunakan pembobotan atribut untuk data kategorik. Data masukan adalah data kategorik hasil ekstraksi fitur dinamis gerakan tangan dari Kinect 2.0. Selain itu juga akan digunakan beberapa dataset lain sebagai pembanding dalam proses uji coba.

Metode ini diharapkan dapat membantu proses klasifikasi terhadap data kategorik yang memiliki dimensi besar, contohnya : data gerakan tangan untuk terjemahan bahasa isyarat.

### 3.1.2 Spesifikasi Kebutuhan Sistem

Proses klasifikasi menggunakan metode k-NN dengan pembobotan atribut, akan dibagi menjadi beberapa tahap. Tahap-tahap tersebut antara lain:

1. Proses masukan data

   Proses masukan data adalah proses membaca data hasil esktraksi fitur dinamis gerakan tangan dalam format *Comma-Separated Values* (.csv).
2. Tahap *Training* (Pembobotan atribut)

Tahap *Training* bertujuan untuk melakukan pembobotan atribut menggunakan metode *Global Weight* dan *Local Weight*.

*3.* Tahap *Testing* (Klasifikasi)
Tahap *Testing* merupakan tahap klasifikasi menggunakan dataset uji coba yang sudah disediakan sebelumnya. Proses klasifikasi juga akan melibatkan bobot masing-masing atribut yang sudah disiapkan pada tahap sebelumnya.

### 3.1.3 Analisis Permasalahan

Dalam proses implementasi terhadap tiap tahap yang ada, tentunya akan terjadi permasalahan yang akan membuat hasil implementasi kurang maksimal. Masalah-masalah tersebut akan dijabarkan sebagai berikut:

#### 3.1.3.1 Analisis Permasalahan Proses Masukan data

Dalam IDE Matlab, semua data yang di masukkan dalam format .csv akan dianggap sebuah matriks numerik. Tentunya hal tersebut dapat menjadi masalah mengingat bahwa data yang digunakan untuk tugas akhir ini bersifat kategorik yang akan dibaca sebagai tipe data *string*.

#### 3.1.3.2 Analisis Permasalahan Tahap *Training*

Dalam tahap *training* akan dilakukan pembobotan atribut menggunakan metode *Global Weight* dan *Local Weight* yang mana masing-masing akan memiliki karakteristik dan cara implementasi yang berbeda-beda. Untuk *Global Weight*, pembobotan dilakukan secara global tanpa mengelompokkan lagi data ke dalam tiap-tiap kelas yang ada. Sedangkan untuk metode *Local Weight* pembobotan akan dilakukan dengan mengacuh terhadap hubungan tiap atribut dengan tiap label kelas yang ada.

#### 3.1.3.3 Analisis Permasalahan Tahap *Testing*

Dalam tahap *testing* akan dilakukan klasifikasi menggunakan metode k-Nearest Neighbour. k-Nearest Neighbour

adalah metode klasifikasi yang simpel. Namun k-Nearest Neighbour yang pada umumnya dirancang untuk data yang bersifat numerik tidak cocok untuk diterapkan terhadap data kategorik yang akan dipakai. Oleh karena itu akan dilakukan beberapa modifikasi terhadap metode klasifikasi tersebut antara lain menggunakan SMC untuk mengukur jarak tiap atribut, dan menggunakan pembobotan yang sudah dihitung pada tahap sebelumnya.

## 3.2 Tahap Perancangan

Tahap perancangan dilakukan untuk merancang proses secara keseluruhan berdasarkan fungsionalitas dan kebutuhan dari klasifikasi k-NN dengan pembobotan atribut untuk data kategorik.

### 3.2.1 Perancangan Sistem

Perancangan sistem dilakukan untuk menggambarkan proses secara keseluruhan dari klasifikasi k-NN dengan pembobotan atribut untuk data kategorik.

Proses di mulai dari pemasukan data berupa tabel data (.csv) yang bersifat kategorik dengan label kelas berada pada kolom terakhir, yang akan di olah ke dalam bentuk matriks dengan ukuran *N*M* di mana *N* adalah jumlah data, dan *M* adalah jumlah atribut + label kelas.

Selanjutnya akan dilakukan proses *training* yang akan terbagi dalam dua tahap. Tahap pertama merupakan tahap perhitungan bobot, menggunakan metode *Global Weight Learning Algorithm* (GWLA) atau *Local Weight Learning Algorithm* (LWLA). Untuk melakukan GWLA atau LWLA, perlu dilakukan perhitungan satuan-satuan *GE*, *GG*, *LE*, dan *LG* yang dapat dihitung menggunakan persamaan **(2.4)**, **(2.6)**, **(2.10)**, dan **(2.12)**. Kemudian akan dilanjutkan pada tahap pemberian bobot untuk tiap-tiap atribut menggunakan bobot global atau lokal menggunakan persamaan **(2.7)**, **(2.9)**, **(2.13)**, dan **(2.14)**.

Tahap yang terakhir adalah tahap uji coba klasifikasi. Pada tahap ini, bobot yang sudah diberikan pada tahap sebelumnya akan

digunakan untuk melakukan klasifikasi terhadap dataset masukan yang sudah disediakan, tentunya pengguna dapat memilih menggunakan beberapa metode pembobotan yang telah dihitung sebelumnya. Proses klasifikasi dilakukan menggunakan metode k-NN dengan WSMC, yang menggunakan SMC sebagai metode perhitungan jarak, kemudian mengalikan jarak tiap atribut data dengan menggunakan bobot yang sudah dihitung pada tahap sebelumnya. Untuk ilustrasi diagram alir dari keseluruhan proses dapat dilihat pada **Gambar 3.1**.

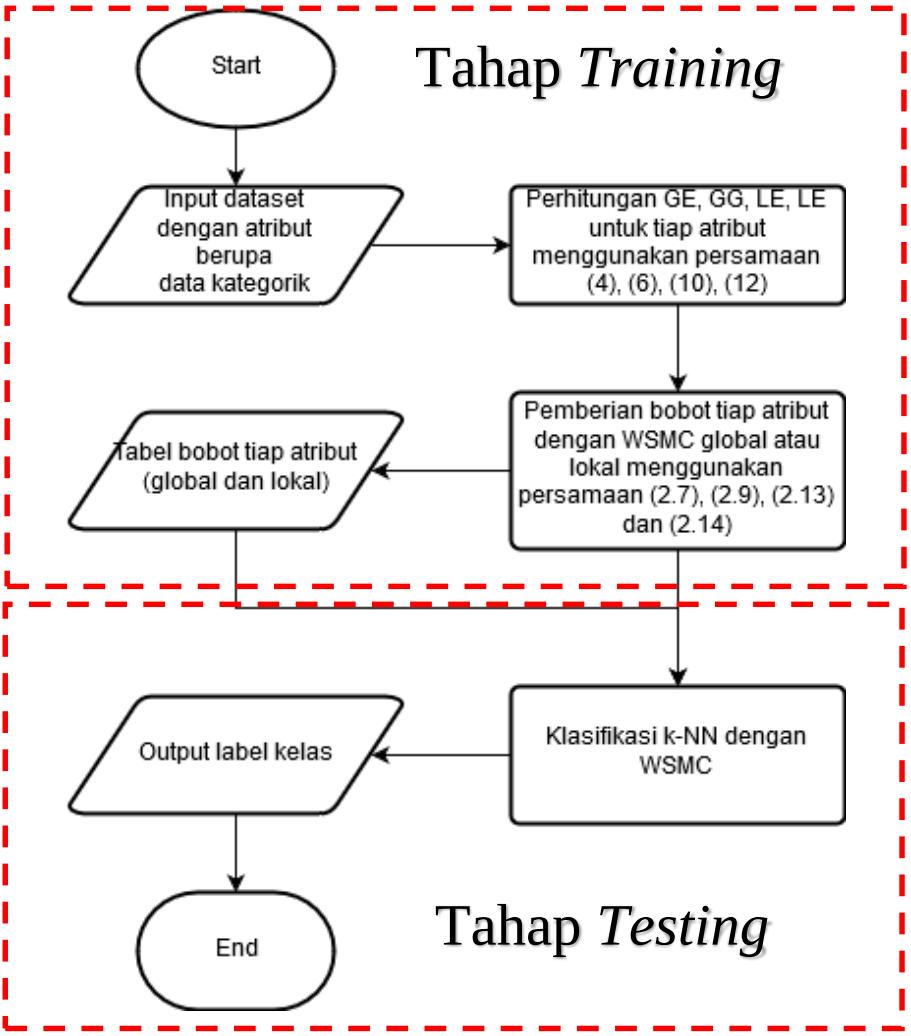
Pada **Gambar 3.1** diagram alir menunjukkan alur proses klasifikasi menggunakan k-NN dengan WSMC secara garis besar. Keluaran dari implementasi klasifikasi k-NN dengan WSMC ini adalah label kelas dari masing-masing data uji coba yang diberikan.

### 3.2.2 Perancangan Data

Perancangan data dilakukan untuk memastikan klasifikasi yang dilakukan berjalan benar dan optimal. Sebagian data masukan adalah data yang digunakan untuk tahap *traning* dan sebagian data yang lain akan digunakan untuk tahap *testing*. Data keluaran adalah data yang ditampilkan kepada pengguna sebagai hasil dari klasifikasi yang telah dilakukan pada tahap *testing*.

Rincian data masukan adalah sebuah tabel berisi sejumlah data dengan atribut bersifat kategorik, disertai label kelas pada bagian kolom terakhir. Data masukan disimpan dalam format .csv.

Data Keluaran adalah berupa label kelas prediksi dari masing-masing data masukan yang digunakan pada tahap *testing*. Selain label kelas prediksi, juga akan ditampilkan akurasi dan label kelas asli dari setiap data yang digunakan dalam tahap *testing*. Data keluaran akan ditampilkan pada antarmuka Matlab IDE dalam bentuk tabel. Contoh format data masukan, dapat dilihat pada **Gambar 3.2**.




**Gambar 3.1** Diagram Alir Implementasi Klasifikasi

Pada **Gambar 3.2** terdapat tujuh buah data tiap barisnya yang dipisahkan oleh tanda koma (,), yang mana data pertama sampai data keenam adalah data atribut yang bersifat kategorik, dan data ketujuh atau data terakhir adalah label kelas. Data pada kolom ke-3 dan ke-4 adalah angka, namun tidak merepresentasikan sebuah data numerik yang perlu untuk dilakukan proses matematika seperti tambah (+), kurang (-), kali (*), bagi (/), melainkan adalah sebuah data yang bersifat kategorik. Jadi angka tersebut hanyalah merepresentasikan nomor kategori saja.

```
vhigh,vhigh,2,2,small,low,unacc
vhigh,vhigh,2,2,small,med,unacc
vhigh,vhigh,2,2,small,high,unacc
vhigh,vhigh,2,2,med,low,unacc
vhigh,vhigh,2,2,med,med,unacc
vhigh,vhigh,2,2,med,high,unacc
vhigh,vhigh,2,2,big,low,unacc
vhigh,vhigh,2,2,big,med,unacc
vhigh,vhigh,2,2,big,high,unacc
vhigh,vhigh,2,4,small,low,unacc
```

**Gambar 3.2** Contoh Format Data di Dalam File cars.csv

### 3.2.3 Perancangan Proses

Perancangan proses dilakukan untuk memberikan gambaran mengenai setiap proses yang dilaksanakan dalam klasifikasi menggunakan k-NN dengan pembobotan atribut.

#### 3.2.3.1 Proses Masukan Data

Pada proses masukan data, data yang didapat akan dimodifikasi terlebih dahulu agar sesuai dengan rancangan data yang sudah direncanakan sebelumnya, yaitu data tabel dengan atribut data kategorik dan label kelas pada kolom terakhir tabel data. Data juga akan disimpan dalam file dengan format .csv.

Setelah data sudah siap untuk diolah, barulah proses masukan data dapat dilakukan melalui Matlab.

Proses masukan data pada Matlab juga akan ada sedikit modifikasi. Mengingat bahwa data yang digunakan adalah data yang bersifat kategorik (kebanyakan bertipe data *string*), maka proses masukan data pada Matlab memerlukan kode tambahan untuk menganggap semua data masukan baik yang bertipe angka maupun *string* akan dianggap sebagai data kategorik (*string*)

### 3.2.3.2 Tahap *Training*

Pada tahap *training*, data masukan yang sudah didapat pada proses sebelumnya akan diproses menggunakan GWLA dan LWLA untuk mendapatkan *Global Weight* dan *Local Weight* untuk tiap atribut. Adapun perhitungan yang digunakan untuk mendapatkan *Global Weight* dan *Local Weight* menggunakan *Entropy* dan *Gini Diversity* yang disebut GE, GG (*Global*) dan LE, LG (*Local*). Segala proses terkait perhitungan yang dilakukan untuk mendapatkan bobot atribut sudah dijabarkan dalam bab sebelumnya pada bagian **Weighted Simple Matching Coefficient (WSMC).**

Pada tahap ini proses dimulai dari perhitungan jumlah atribut dan kategori kelas dari data masukan yang akan digunakan untuk proses perhitungan GE, GG, LE, dan LG. Selanjutnya dilanjutkan dengan perhitungan GE, GG, LE dan LG menggunakan persamaan **(2.4)**, **(2.6)**, **(2.10)**, dan **(2.12)**. Setelah didapatkan masing-masing nilai GE, GG, LE dan LG, dilanjutkan dengan proses perhitungan bobot global dan lokal untuk masing-masing metode perhitungan menggunakan persamaan **(2.7)**, **(2.9)**, **(2.13)**, dan **(2.14)**. Keluaran adalah berupa tabel bobot atribut dari masing-masing metode perhitungan, yang nantinya akan dipakai pada proses *testing.* Diagram alir untuk tahap *training* ditunjukkan pada **Gambar 3.3**.

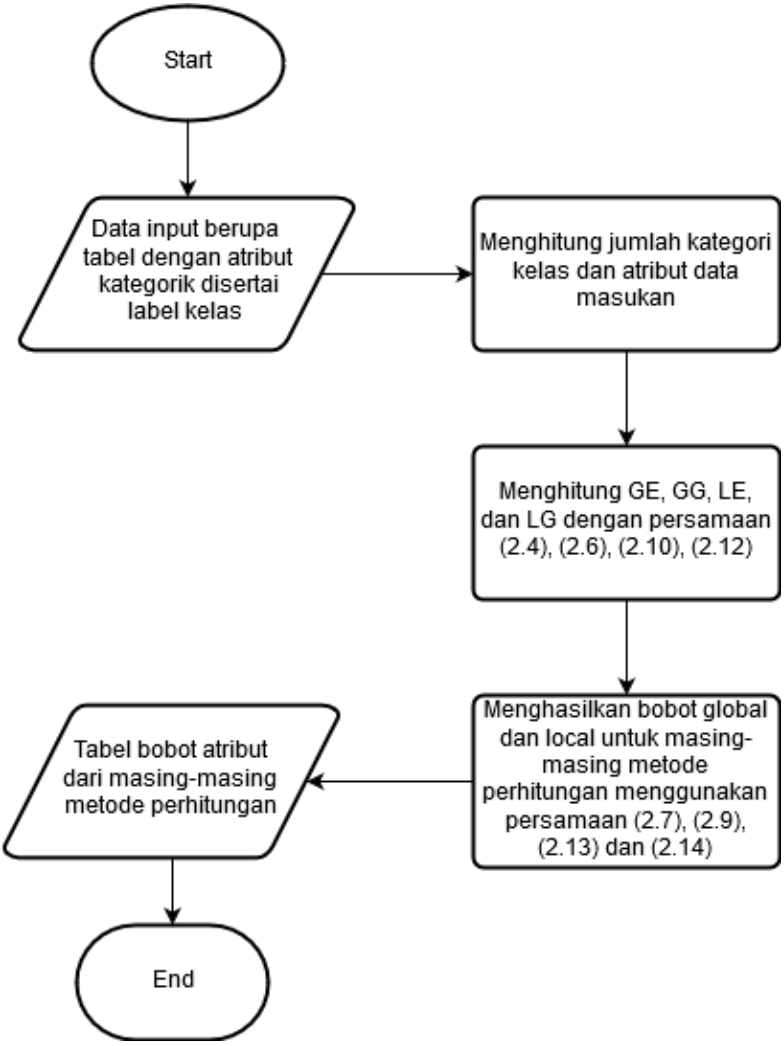
#### 3.2.3.3 Tahap *Testing*

Tahap *testing* adalah tahap klasifikasi untuk dataset yang diberikan. Dengan bantuan dari bobot yang sudah dihitung pada tahap *training*, diharapkan klasifikasi dapat menjadi lebih optimal dan akurat. Metode klasifikasi yang digunakan adalah k-NN dengan pembobotan atribut, yang sudah dijabarkan pada bab sebelumnya pada **k-Nearest Neighbor dengan WSMC.**

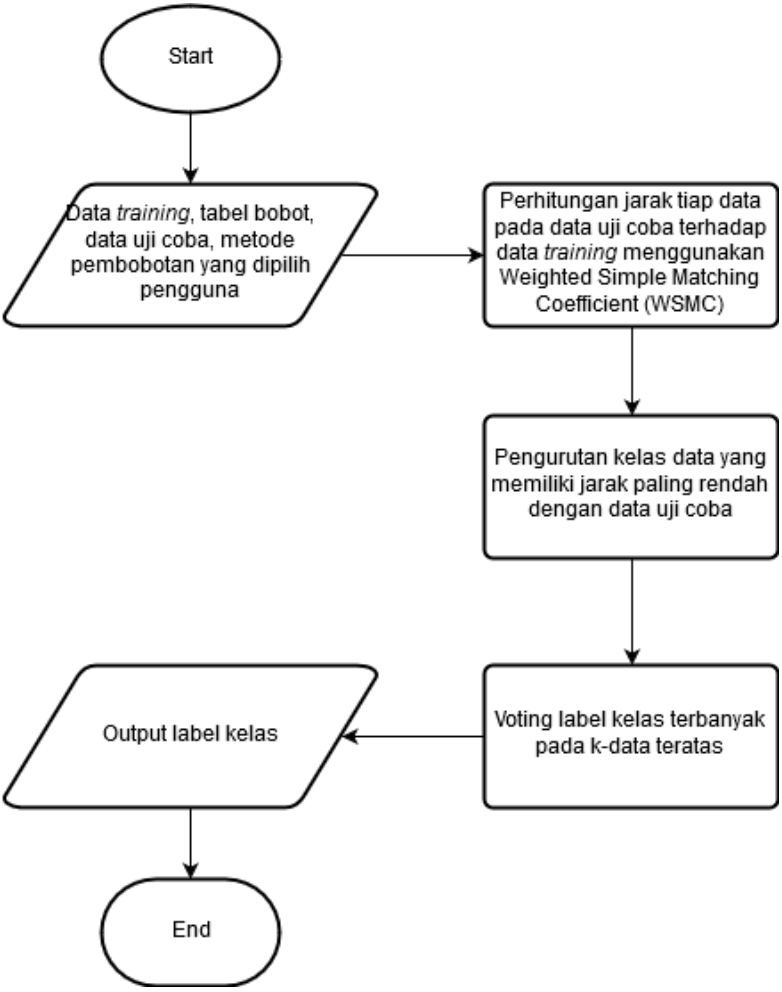
Pada tahap ini proses dimulai dari data masukan berupa tabel bobot dan data *training* dari tahap *training*. Pengguna juga diminta untuk memasukkan dataset sebagai data *testing*. Proses dilanjutkan dengan melakukan pengecekan terhadap data *testing*. Data *testing* dianggap sesuai dengan data *training* apabila memiliki jumlah atribut yang sama dengan jumlah atribut pada data training. Apabila data *testing* sesuai dengan data *training*, maka proses dilanjutkan ke tahap pemilihan metode perhitungan. Pengguna dapat memilih metode perhitungan bobot yang mana yang akan digunakan. Terdapat empat pilihan perhitungan bobot yaitu GE, GG, LE, dan LG. Proses dilanjutkan dengan klasifikasi menggunakan k-NN dengan WSMC berdasarkan metode pembobotan yang sudah dipilih (diagram alir dapat dilihat pada **Gambar 3.4**). Keluaran dari proses ini adalah tabel data dengan label kelas prediksi dan perhitungan akurasi yang akan ditampilkan pada antarmuka Matlab IDE. Adapun diagram alir tahap testing dapat dilihat pada **Gambar 3.5.**

Untuk proses klasifikasi k-NN yang ditunjukkan pada **Gambar 3.4**, proses dimulai setelah pengguna memilh metode pembobotan yang disediakan. Setelah itu tiap data pada data uji coba akan dihitung jaraknya menggunakan WSMC, dimana setiap atribut pada data uji coba akan dibandingkan dengan atribut pada data training, dan dikalikan dengan bobot dari tabel bobot yang tersedia. Proses dilanjutkan dengan pengurutan data yang memiliki jarak paling rendah dengan data uji coba. Kemudian akan diambil k-data teratas (jarak paling minimum) untuk melakukan voting label kelas. Label kelas dengan suara terbanyak akan dinyatakan

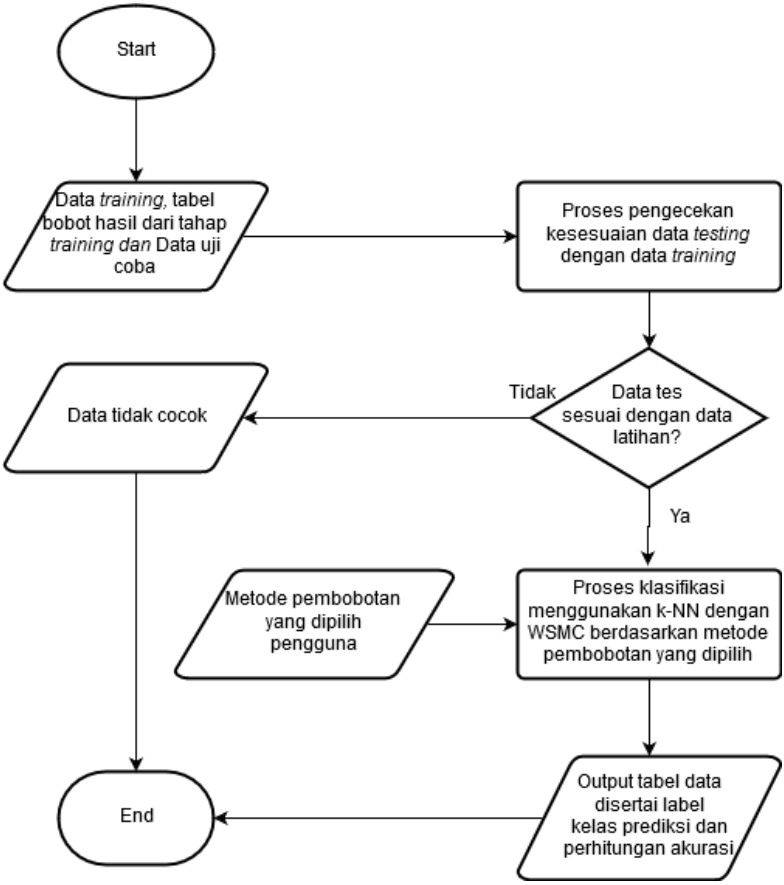
sebagai label kelas dari data uji coba dan akan dijadikan sebagai keluaran pada tahap ini.




**Gambar 3.3** Diagram Alir Tahap *Training*




**Gambar 3.4** Diagram Alir Klasifikasi k-NN dengan WSMC




**Gambar 3.5** Diagram Alir Tahap *Testing*

**[*Halaman ini sengaja dikosongkan*]**

# BAB IV
# IMPLEMENTASI

Pada bab ini diuraikan mengenai implementasi klasifikasi dari rancangan metode yang telah dibahas pada Bab III meliputi kode program dalam perangkat lunak. Selain itu, implementasi dari tiap proses, parameter masukan, keluaran, dan beberapa keterangan yang berhubungan dengan program juga dijelaskan.

## 4.1 Lingkungan Implementasi

Dalam implementasi algoritma klasifikasi data kategorik, digunakan perangkat-perangkat sebagai berikut:

### 4.1.1 Perangkat Keras

Lingkungan implementasi pada tugas akhir ini adalah sebuah Laptop. Perangkat laptop yang digunakan adalah ASUS - K46CB dengan ukuran layar 14 inch. Spesifikasi dari PC yang digunakan pada tugas akhir ini adalah:

- Processor : Intel(R) Core(TM) i5-3317U CPU @ 1.70GHz
- RAM : 4.00 GB (3.89 GB *usable*)
- OS : Windows 8.1 Enterprise N 64-bit

### 4.1.2 Perangkat Lunak

Lingkungan implementasi pada tugas akhir ini adalah sebuah Laptop. Software yang digunakan untuk implementasi kode adalah Matlab R2015a 64-bit dengan dibantu IDE dari Matlab IDE.

Selain itu, pada tugas akhir ini dalam melakukan pengolahan data input didukung dengan *software Microsoft Excel* dan *Notepad++*.

## 4.2 Implementasi Kode

Implementasi kode dalam tugas akhir ini terbagi menjadi tiga tahap, tahap masukan data, tahap *training* dan tahap *testing*.

Implementasi dari masing-masing tahap akan dijabarkan sebagai berikut:

### 4.2.1 Implementasi Tahap Masukan Data

Tahap ini adalah tahap membaca file berbentuk .csv ke dalam Matlab. Proses pembacaan file .csv dalam Matlab, akan otomatis terkonversi menjadi numerik. Apabila dimasukkan data berbentuk kategorik, matlab akan menganggap hal tersebut sebagai error, maka dari itu dibutuhkan kode tambahan untuk membaca data masukan berbentuk .csv. Implementasi kode untuk membaca file di dalam matlab dapat dilihat pada **Kode Sumber 4.1**. Sedangkan implementasi kode fungsi untuk membaca file .csv yang berjenis data kategorik dapat dilihat pada **Kode Sumber 4.2.** Hasil pembacaan file ke dalam matlab dapat dilihat contohnya pada **Gambar 4.1.**

Pada **Kode Sumber 4.1** fungsi *read_mixed_csv* dipanggil dan dimasukkan ke variabel penampung bernama *data*. Sehingga seluruh isi dari variabel *lineArray* yang dideklarasikan pada **Kode Sumber 4.2**, akan dipindahkan ke dalam variabel *data*. Variabel *lineArray* adalah variabel penampung bertipe *Cell Array* yang akan menampung hasil keluaran dari fungsi *read_mixed_csv*. Fungsi pada **Kode Sumber 4.2** akan terus berjalan sampai seluruh file yang dibaca habis (sampai pada baris terakhir). Dalam proses pembacaaan data yang dilakukan per baris, data yang ada dibaca menggunakan tipe data *string*, sehingga didapatkan keluaran bertipe *Cell Array*, dengan isi data bertipe *string*.

```
1  [FileName,PathName] = uigetfile('*.*','Select
2  Data set file for Training Phase');
3      PathFileName = strcat(PathName,FileName);
4      data = read_mixed_csv(PathFileName,',');
```

**Kode Sumber 4.1** Pembacaan File ke Dalam Matlab

```
function lineArray =
read_mixed_csv(fileName,delimiter)
  fid = fopen(fileName,'r');
  lineArray = cell(100,1);
  lineIndex = 1;
  nextLine = fgetl(fid);
  while ~isequal(nextLine,-1)
    lineArray{lineIndex} = nextLine;
    lineIndex = lineIndex+1;
    nextLine = fgetl(fid);
  end
  fclose(fid);
  lineArray = lineArray(1:lineIndex-1);
  for iLine = 1:lineIndex-1
    lineData =
textscan(lineArray{iLine},'%s',...
'Delimiter',delimiter);
    lineData = lineData{1};
    if strcmp(lineArray{iLine}(end),delimiter)
      lineData{end+1} = '';
    end
    lineArray(iLine,1:numel(lineData)) =
lineData;
  end
end
```

**Kode Sumber 4.2** Pembacaan File .csv yang Mengandung String

### 4.2.2 Implementasi Tahap *Training*

Dalam tahap *training*, proses dimulai dari inisiasi data dengan mendefinisikan jumlah data, jumlah kategori atribut, jumlah label kelas dan lain sebagainya. Dilanjutkan dengan mencari perhitungan-perhitungan lain seperti mencari *GE*, *GG*, $\omega_d^{(GE)}$, $\omega_d^{(GG)}$ (*Global Weights*) dan *LG*, *LE*, $W_{md}^{(LE)}$, $W_{md}^{(LG)}$ (*Local Weights*).

Implementasi kode proses inisiasi data dapat dilihat pada **Kode Sumber 4.3, Kode Sumber 4.4, Kode Sumber 4.5,** dan **Kode Sumber .**

data

1557x7 cell

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 1 | vhigh | med | 2 | 4 | small | high | acc |
| 2 | vhigh | med | 2 | 4 | med | high | acc |
| 3 | vhigh | med | 2 | 4 | big | med | acc |
| 4 | vhigh | med | 2 | 4 | big | high | acc |
| 5 | vhigh | med | 2 | more | med | high | acc |
| 6 | vhigh | med | 2 | more | big | med | acc |
| 7 | vhigh | med | 2 | more | big | high | acc |
| 8 | vhigh | med | 3 | 4 | small | high | acc |
| 9 | vhigh | med | 3 | 4 | med | high | acc |
| 10 | vhigh | med | 3 | 4 | big | med | acc |
| 11 | vhigh | med | 3 | 4 | big | high | acc |
| 12 | vhigh | med | 3 | more | small | high | acc |
| 13 | vhigh | med | 3 | more | med | med | acc |
| 14 | vhigh | med | 3 | more | med | high | acc |
| 15 | vhigh | med | 3 | more | big | med | acc |
| 16 | vhigh | med | 3 | more | big | high | acc |

**Gambar 4.1** Contoh Hasil Pembacaan File .csv dalam Matlab

```
1  N = size(data,1);
2  D = size(data,2) - 1;
```

**Kode Sumber 4.3** Inisiasi Jumlah Data dan Jumlah Atribut

Pada **Kode Sumber 4.3** inisiasi jumlah data dan jumlah atribut, dengan *N* adalah jumlah data, dan *D* adalah jumlah atribut data.

```
1  for i=1:D
2      SDcount(i) = 0;
3      for j=1:N
4          if ~any(strcmp(SDtemp,data{j,i}))
5              SDtemp{1,SDcount(i)+1} =
6                  data{j,i};
7              SD{i,SDcount(i)+1} = data{j,i};
8              SDcount(i) = SDcount(i)+1;
9          end
10     end
11      SDtemp = {}
12 end
```

**Kode Sumber 4.4** Pendefinisian dan Perhitungan Kategori Atribut

Pada **Kode Sumber 4.4** *SDCount* adalah variabel berupa *cell array* yang menyimpan jumlah kategori atribut untuk tiap kolom atribut dan *SD* adalah variabel berupa *cell array* yang menyimpan apa saja kategori dari tiap atribut yang ada.

```
1  classcount = 0;
2  for i=1:N
3      if ~any(strcmp(class_list,data{i,D+1}))
4              class_list{1, classcount+1} =
5                  data{i,D+1};
6              classcount = classcount+1;
7      end
8  end
```

**Kode Sumber 4.5** Perhitungan Jumlah Kategori Label Kelas

Pada **Kode Sumber 4.5** *classcount* adalah variabel berupa *integer* yang menyimpan jumlah kategori label kelas, dan *class_list* adalah variabel berupa *cell array* yang menyimpan kategori apa saja yang dimiliki oleh label kelas.

**Kode Sumber 4.3, Kode Sumber 4.4, Kode Sumber 4.5** mendefinisikan data-data atau variabel yang dibutuhkan dalam proses perhitungan untuk mencari *Global Weight* yang dapat dilihat pada **Kode Sumber 4.6.**

Pada **Kode Sumber 4.6** *GESD* dan *GGSD* adalah variabel berupa *float* yang menyimpan nilai *GE* dan *GG* yang dihitung berdasarkan pengaruh tiap kategori atribut dengan masing-masing label kelas. Pada baris ke 31 sampai 33, dilakukan proses rata-rata untuk tiap nilai *GE* dan *GG* yang didapatkan dari masing-masing label kelas, sehingga bobot yang dihasilkan hanya 1 bobot untuk masing-masing nomor atribut. Persamaan yang digunakan dalam implementasi perhitungan *Global Weight* dapat dilihat pada bagian sebelumnya pada subbab **Weighted Simple Matching Coefficient (WSMC)**. Adapun fungsi *pmsd* yang membantu dalam perhitungan *Global Weight* dapat dilihat kode sumbernya pada **Kode Sumber 4.7.**

```
d=size(SD,2);
M=size(class_list,2);
for j=1:D
    AGE = 0;
    AGG = 0;
    for k=1:d
        GESD = 0;
        sumGGSD = 0;
        for i=1:M
            calculatePMSD1 =
                pmsd(data,class_list{1,i},
                SD{j,k},j,N);
            if calculatePMSD1==0
            else
                GESD = GESD +
                    calculatePMSD1*
                    log2(calculatePMSD1);
                sumGGSD = sumGGSD +
                calculatePMSD1*
                    calculatePMSD1;
            end
        end
        GGSD = 1 - sumGGSD;
        GESD = GESD*-1;
        PSD = 0;
        for i=1:N
            if strcmp(data{i,j},SD{j,k})
                PSD = PSD + 1;
            end
        end
        PSD = PSD/N;
        AGE = AGE + PSD*GESD;
        AGG = AGG + PSD*GGSD;
    end
    AGE = AGE/log2(M);
    AGG = AGG*M/(M-1);
    WGE(j) = exp(-1*AGE);
     WGG(j) = exp(-1*AGG);
end
```

**Kode Sumber 4.6** Perhitungan *Global Weight*

```
function calculate = pmsd(data,m,sd,D,N)
count_cm = 0;
count_tr = 0;
for i=1:N
    if strcmp(data{i,D},sd)
        if strcmp(data{i,end},m)
            count_cm = count_cm +1;
            count_tr = count_tr +1;
        else
            count_tr = count_tr +1;
        end
    end
end
if count_tr == 0
    calculate = 0;
else
    calculate = count_cm/count_tr;
end
end
```

**Kode Sumber 4.7** Fungsi *pmsd*

Pada **Kode Sumber 4.7** *m* dan *sd* adalah variabel berupa *string* yang masing-masing secara berturut-turut menyimpan sebuah kategori label kelas dan kategori atribut. *D* adalah nomor atribut yang disoroti, *count_cm* adalah variabel berupa *integer* yang menyimpan jumlah data yang memiliki atribut *sd* pada nomor atribut *D* dan memiliki label kelas *m*. Sedangkan *count_tr* hanya menyimpan jumlah data yang memiliki atribut *sd* pada nomor atribut *D* tidak peduli apapun label kelasnya. Dari fungsi *pmsd* ini, dapat diketahui sejauh mana pengaruh atribut *sd* pada nomor atribut *D* terhadap kategori label kelas *m*.

```
1  for j=1:M
2      class_counter(j) = 0;
3      for i=1:N
4          if strcmp(class_list{1,j},data{i,end})
5              class_counter(j) =
6                  class_counter(j)+1;
7          end
8      end
9  end
```

**Kode Sumber** Perhitungan Jumlah Data yang Dimiliki tiap Label Kelas

Pada **Kode Sumber** *class_counter* adalah variabel berupa *cell array* yang menyimpan jumlah data yang dimiliki oleh tiap label kelas. Variabel *class_counter* adalah data tambahan yang diperlukan untuk menghitung *Local Weight* yang implementasi kodenya dapat dilihat pada **Kode Sumber 4.8.**

Pada **Kode Sumber 4.8** *sumLE* dan *sumLG* adalah variabel berupa *float* yang menyimpan nilai *LE* dan *LG*. Berbeda dengan *Global Weight*, pada perhitungan untuk mencari *Local Weight*, tidak dilakukan proses rata-rata untuk masing-masing kategori label kelas. Sehingga bobot yang dihasilkan memiliki dimensi sebanyak *M***D* dengan *M* adalah jumlah kategori label kelas, dan *D* adalah jumlah atribut. Adapun fungsi *psdm* yang membantu dalam proses perhitungan bobot *Local Weight*, kode sumbernya dapat dilihat pada **Kode Sumber 4.9.**

Pada **Kode Sumber 4.9** fungsi *psdm* membantu menghitung pengaruh tiap kategori atribut pada nomor atribut *d* terhadap kategori label kelas pada data. Contoh data keluaran dari tahap *training* dapat dilihat pada **Gambar 4.2.**

Pada **Gambar 4.2** dapat dilihat bahwa bobot untuk metode global hanya berjumlah dua baris, yaitu *WeightGE* dan *WeightGG*. Sedangkan bobot untuk metode lokal memiliki jumlah baris lebih banyak yaitu sejumlah dengan label kelas yang ada.

| | Attribue #1 | Attribue #2 | Attribue #3 | Attribue #4 | Attribue #5 | Attribue #6 |
|---|---|---|---|---|---|---|
| WeightGE | 0.5831 | 0.5685 | 0.5481 | 0.6075 | 0.5560 | 0.6225 |
| WeightGG | 0.5584 | 0.5526 | 0.5441 | 0.5946 | 0.5487 | 0.6019 |
| WeightLEC1 | 0.3808 | 0.3704 | 0.3687 | 0.5323 | 0.3753 | 0.5381 |
| WeightLEC2 | 0.6371 | 0.6271 | 0.3688 | 0.5336 | 0.3713 | 0.5396 |
| WeightLEC3 | 0.3825 | 0.3729 | 0.3685 | 0.3801 | 0.3692 | 0.3824 |
| WeightLEC4 | 0.6301 | 0.4651 | 0.3841 | 0.5338 | 0.5441 | 1 |
| WeightLGC1 | 0.3793 | 0.3702 | 0.3686 | 0.4727 | 0.3757 | 0.4811 |
| WeightLGC2 | 0.5611 | 0.5455 | 0.3687 | 0.4746 | 0.3715 | 0.4833 |
| WeightLGC3 | 0.3799 | 0.3727 | 0.3684 | 0.3820 | 0.3694 | 0.3842 |
| WeightLGC4 | 0.5502 | 0.4247 | 0.3821 | 0.4749 | 0.4899 | 1 |

**Gambar 4.2** Contoh Tabel Keluaran Tahap Training

```
for i=1:M
    for j=1:D
        sumLE = 0;
        sumLG = 0;
        for k=1:d
            calculatePSDM =
                psdm(data,SD{j,k},
                    class_list{1,i},
                    class_counter(i),j,N);
            if calculatePSDM~=0
                sumLE = sumLE+(calculatePSDM*
                    log2(calculatePSDM));
                sumLG = sumLG+(calculatePSDM*
                    calculatePSDM);
            end
        end
        sumLE = -1*sumLE;
        sumLG = 1 - sumLG;
        WLE(i,j) = exp(-1*sumLE/
            log2(SDcount(j)));
        WLG(i,j) = exp(-1*sumLG*SDcount(j)/
            (SDcount(j)-1));
    end
end
```

**Kode Sumber 4.8** Perhitungan *Local Weight*

```
function calculate = psdm(data, sd, cm, ncm, d,
N)
    count_total = 0;
    for i=1:N
        if strcmp(data{i,end},cm)
            if strcmp(data{i,d},sd)
                count_total = count_total+1;
            end
        end
    end
    count_total = count_total/ncm;
    calculate = count_total;
end
```

**Kode Sumber 4.9** Fungsi *psdm*

### 4.2.3 Implementasi Tahap *Testing*

Pada tahap *testing*, proses dimulai dari pengecekan data yang dimasukkan oleh pengguna. Sistem akan melakukan pengecekan apakah data sesuai dengan data *training* atau tidak. Jika data cocok, akan dilanjutkan dengan proses pemilihan metode pembobotan, dan dilanjutkan proses klasifikasi.

Kode sumber untuk pengecekan data masukan dapat dilihat pada **Kode Sumber 4.10**.

```
[FileNameTest,PathNameTest] =
    uigetfile('*.*', '');
PathFileNameTest =
    strcat(PathNameTest,FileNameTest);
data_test =
    read_mixed_csv(PathFileNameTest,',');
if size(data_test,2)-1~=D
    alertBox =
        msgbox('Data Test not Matched!',
        'ERROR!!!','error');
end
```

**Kode Sumber 4.10** Pengecekan Kecocokan Data

Pada **Kode Sumber 4.10** dilakukan pengecekan terhadap jumlah atribut. Apabila jumlah atribut data uji coba tidak sama dengan jumlah atribut pada data *training*, maka data dianggap tidak cocok. Proses selanjutnya dapat dilihat pada **Kode Sumber 4.11**.

```
selectionStringTestInput = {'Global
    Entropy','Global Gini Diversity','Local
    Entropy','Local Gini Diversity'};
[s_answer_data_test,v_answer_data_test] =
listdlg('PromptString','Select weighting method
    for testing:',...
    'SelectionMode','single',...
    'ListSize',[300 100],...
    'ListString',selectionStringTestInput);
```

**Kode Sumber 4.11** Pemilihan Metode Pembobotan

```
for i=1:data_test_N
    distance = {};
     for j=1:N
         distance_count = 0;
         for k=1:D
             if ~strcmp(data_test{i,k}
                 ,data{j,k}
                 distance_count =
                     distance_count +
                     1*WGE(k);
             end
         end
         distance{j,1} = distance_count;
         distance{j,2} = data{j,D+1};
     end
     out = sortrows(distance,1);
     temp_class = {};
     temp_class_count = 0;
     for x=1:KNN
        if ~any(strcmp(temp_class,out{x,2}))
             temp_class{temp_class_count+1,
                 1} = out{x,2};
             temp_class{temp_class_count+1,
                 2} = 1;
             temp_class_count =
```

```
26                      temp_class_count+1;
27              else
28                   temp_class{temp_class_count,2} =
29                      temp_class{temp_class_count,2}
30                      + 1;
31              end
32          end
33          out = sortrows(temp_class,2);
34          data_test_class{i} = out{1,1};
35          if strcmp(out{1,1},data_test{i,D+1})
36              count_correct = count_correct+1;
37          else
38              count_false = count_false+1;
39          end
40      end
```

**Kode Sumber 4.12** Klasifikasi k-NN dengan WSMC

Pada **Kode Sumber 4.11** pengguna diminta untuk memilih metode pembobotan yang disediakan. Setelah metode pembobotan dipilih, proses dilanjutkan pada tahap klasifikasi. Kode sumber proses klasifikasi dapat dilihat pada **Kode Sumber 4.12.**

Pada **Kode Sumber 4.12** baris 3 sampai baris 15 adalah kode untuk menghitung jarak setiap data tes dengan data *training*, sesuai dengan metode pembobotan yang dipilih (*WGE(k)* dapat diganti dengan *WGG*, *WLE*, atau *WLG* tergantung metode pembobotan yang dipilih). Selanjutnya pada baris 19 sampai 34, ditentukan label kelas tiap data tes berdasarkan jarak paling minimum terhadap data *training*. Pada baris 35 sampai 39 adalah proses perhitungan akurasi dibandingkan dengan kelas label yang asli. Contoh keluaran dari tahap *testing* ditampilkan dalam bentuk *screenshot figure* Matlab, yang dapat dilihat pada **Gambar 4.3.**

Testing Result

Number of Instances : 171
Number of Attributes : 6
Class Category : 4
Weighting Method :Global || Entropy

Correct Result : 133
False Result : 38
Accuracy : 77.78 %

| | Attribue #1 | Attribue #2 | Attribue #3 | Attribue #4 | Attribue #5 | Attribue #6 | Original Class | Resu |
|---|---|---|---|---|---|---|---|---|
| 1 | low | high | 3 | more | med | med | acc | acc |
| 2 | low | high | 3 | more | big | med | acc | acc |
| 3 | low | high | 4 | 4 | small | med | acc | unacc |
| 4 | low | high | 4 | 4 | small | high | acc | vgood |
| 5 | low | high | 4 | 4 | med | med | acc | acc |
| 6 | low | high | 4 | 4 | big | med | acc | acc |
| 7 | low | high | 4 | more | small | med | acc | acc |
| 8 | low | high | 4 | more | small | high | acc | acc |
| 9 | low | high | 4 | more | med | med | acc | good |
| 10 | low | high | 4 | more | big | med | acc | good |
| 11 | low | high | 5more | 4 | small | med | acc | unacc |
| 12 | low | high | 5more | 4 | small | high | acc | vgood |

**Gambar 4.3** Contoh Keluaran Tahap Testing

**[*Halaman ini sengaja dikosongkan*]**

# BAB V
# UJI COBA DAN EVALUASI

Dalam bab ini dibahas mengenai hasil uji coba sistem yang telah dirancang dan dibuat. Uji coba dilakukan untuk mengetahui kinerja sistem dengan lingkungan uji coba yang telah ditentukan.

## 5.1 Lingkungan Uji Coba

Lingkungan uji coba pada tugas akhir ini adalah sebuah Laptop. Perangkat laptop yang digunakan adalah ASUS - K46CB dengan ukuran layar 14 inch. Spesifikasi dari PC yang digunakan pada tugas akhir ini adalah:

- Processor : Intel(R) Core(TM) i5-3317U CPU @ 1.70GHz
- RAM : 4.00 GB (3.89 GB *usable*)
- OS : Windows 8.1 Enterprise N 64-bit

Software yang digunakan untuk uji coba adalah Matlab R2015a 64-bit dengan dibantu IDE dari Matlab IDE. Selain itu, pada tugas akhir ini dalam melakukan pengolahan data input didukung dengan *software Microsoft Excel* dan *Notepad++*. Untuk membandingkan data hasil uji coba, juga digunakan *software* pembantu WEKA versi 3.6.

## 5.2 Data Uji Coba

Data uji coba yang digunakan sebagai masukan adalah data tabel dalam format .csv. Adapun isi dari tabel adalah atribut yang bersifat kategorik, diikuti dengan label kelas pada kolom terakhir. Jumlah data, jumlah atribut, jumlah kategori atribut dan jumlah kategori label kelas pada dataset uji coba bervariatif untuk melakukan uji coba karakteristik masing-masing metode pembobotan.

Untuk data yang digunakan serta parameter jumlah atribut dan jumlah kategori label kelas dapat dilihat pada **Tabel 5.1**. Adapun

untuk dataset selain data gerakan tangan didapatkan dari *http://ftp.ics.uci.edu/pub/machine-learning-databases/*.

**Tabel 5.1** Data yang Digunakan untuk Uji Coba

| Data Set | #Atribut (D) | Contoh nilai atribut | #Kelas (M) | #Jumlah Data (N) |
|---|---|---|---|---|
| *Nursery* | 8 | *Recommended, priority, not_recom* | 5 | 12958 |
| *Cars* | 6 | *Vgood, good, 5more* | 4 | 1728 |
| Gerakan Tangan | 36 | 6, 7, 8 (kategorik) | 10 | 150 |
| *Soybeans* | 35 | 3, 0, 2 (kategorik) | 19 | 683 |
| *Vote* | 16 | Y, N | 2 | 435 |
| *Dhermatology* | 34 | 0, 1, 2 (kategorik) | 6 | 366 |

## 5.3 Skenario Uji Coba

Uji coba dilakukan untuk membandingkan tingkat ketepatan (akurasi) dari metode yang dipakai, dibandingkan dengan metode lainnya yang sudah ada sebelumnya.

Skenario uji coba adalah melakukan uji coba terhadap enam dataset yaitu *Nursery*, *Cars*, Gerakan Tangan, *Soybeans*, *Dhermatology*, dan *Vote*. Nantinya dataset akan diuji dengan menggunakan metode k-NN dengan WSMC baik itu global, maupun lokal, dan akan dilakukan uji coba dengan k-NN biasa dengan bantuan WEKA. Masing-masing klasifikasi k-NN akan ditentukan nilai *k* nya, yaitu 3 baik untuk k-NN dengan WSMC maupun k-NN dengan bantuan WEKA. Data uji coba akan diambil dari sepersepuluh dari total jumlah data dengan persebaran label

kelas yang merata. Adapun tabel uji coba terhadap setiap dataset yang belum dirata-rata dapat dilihat pada bagian **LAMPIRAN.**

## 5.4 Uji Coba Pada Data *Nursery*

Dataset *Nursery* adalah dataset yang paling banyak jumlah datanya. Dengan persebaran data untuk tiap label kelas berkisar 4 ribu data untuk 3 kelas, dan berkisar tiga ratus data untuk 1 label kelas. Sedangkan untuk 1 kelas yang lain hanya mendapat 2 data saja. Data akan diuji coba dengan metode *k-Fold Cross Validation* dengan nilai k adalah 10. Untuk hasil uji coba akurasi rata-rata dapat dilihat pada **Tabel 5.2**.

**Tabel 5.2** Uji Coba Data *Nursery*

| Metode | Jumlah Data Rata-rata | Prediksi Benar Rata-rata | Prediksi Salah Rata-rata | Akurasi Rata-rata (%) |
|---|---|---|---|---|
| Global Entropy | 1296 | 939 | 357 | 72.45 |
| Global Gini Diversity | | 943 | 353 | 72.76 |
| Local Entropy | | 997 | 299 | 76.92 |
| Local Gini Diversity | | 1002 | 294 | 77.31 |
| k-NN SMC (Matlab) | | 519 | 777 | 40.04 |
| k-NN (WEKA) | | 990 | 306 | 76.38 |

Pada **Tabel 5.2** terlihat bahwa metode yang diajukan baik untuk metode GE, GG, LE dan LG, semua menunjukkan akurasi di atas k-NN biasa tanpa pembobotan atribut (Matlab). Baris yang

diberi warna kuning menandakan metode dengan tingkat akurasi paling tinggi.

## 5.5 Uji Coba Data *Cars*

Dataset *Cars* adalah dataset yang menengah jumlah datanya. Dengan persebaran data untuk tiap label kelas berkisar 1200 data untuk 1 kelas, dan berkisar tiga ratus data untuk 1 label kelas. Sedangkan untuk 2 kelas yang lain hanya mendapat sekitar 60 data saja. Data akan diuji coba dengan metode *k-Fold Cross Validation* dengan nilai k adalah 10. Untuk hasil uji coba akurasi rata-rata dapat dilihat pada **Tabel 5.3.**

**Tabel 5.3** Uji Coba Data *Cars*

| Metode | Jumlah Data Rata-rata | Prediksi Benar Rata-rata | Prediksi Salah Rata-rata | Akurasi Rata-rata (%) |
|---|---|---|---|---|
| Global Entropy | 173 | 134 | 39 | 77.46 |
| Global Gini Diversity | | 134 | 39 | 77.46 |
| Local Entropy | | 139 | 34 | 80.35 |
| Local Gini Diversity | | 138 | 35 | 79.77 |
| k-NN SMC (Matlab) | | 112 | 61 | 64.74 |
| k-NN (WEKA) | | 129 | 44 | 74.77 |

Pada **Tabel 5.3** terlihat bahwa metode yang diajukan untuk LE dapat mendapatkan akurasi yang lebih baik dari metode global maupun k-NN tanpa pembobotan atribut (Matlab). Baris yang

diberi warna kuning menandakan metode dengan tingkat akurasi paling tinggi.

## 5.6 Uji Coba Data Gerakan Tangan

Dataset Gerakan Tangan adalah dataset yang sedikit jumlah datanya. Dengan persebaran data untuk tiap label kelas berkisar 15 data untuk tiap kelas. Data akan diuji coba dengan metode *k-Fold Cross Validation* dengan nilai k adalah 10. Untuk hasil uji coba akurasi rata-rata dapat dilihat pada **Tabel 5.4**.

**Tabel 5.4** Uji Coba Data Gerakan Tangan

| Metode | Jumlah Data Rata-rata | Prediksi Benar Rata-rata | Prediksi Salah Rata-rata | Akurasi Rata-rata (%) |
|---|---|---|---|---|
| Global Entropy | 20 | 18 | 2 | 90.50 |
| Global Gini Diversity | | 18 | 2 | 90.50 |
| Local Entropy | | 19 | 1 | 93.50 |
| Local Gini Diversity | | 19 | 1 | 97.00 |
| k-NN SMC (Matlab) | | 18 | 2 | 92.00 |
| k-NN (WEKA) | | 19 | 1 | 95.50 |

Pada **Tabel 5.4** terlihat bahwa metode LG memiliki akurasi rata-rata tertinggi dari semua metode. Baris yang diberi warna kuning menandakan metode dengan tingkat akurasi paling tinggi.

## 5.7 Uji Coba Data *Soybeans*

Dataset *Soybeans* adalah dataset yang menengah jumlah datanya. Namun data ini memiliki jumlah atribut yang cukup banyak, serta kategori label kelas yang mencapai 19 label kelas. Persebaran kelas pada data *Soybeans* dapat dikatakan cukup merata. Data akan diuji coba dengan metode *k-Fold Cross Validation* dengan nilai k adalah 10. Untuk hasil uji coba akurasi rata-rata dapat dilihat pada **Tabel 5.5.**

Pada **Tabel 5.5** metode Lokal dengan LE, mendapatkan akurasi paling tinggi dibandingkan dengan metode global dan k-NN tanpa pembobotan atribut. Baris yang diberi warna kuning menandakan metode dengan tingkat akurasi paling tinggi.

**Tabel 5.5** Uji Coba Data *Soybeans*

| Metode | Jumlah Data Rata-rata | Prediksi Benar Rata-rata | Prediksi Salah Rata-rata | Akurasi Rata-rata (%) |
|---|---|---|---|---|
| Global Entropy | 68 | 61 | 7 | 89.31 |
| Global Gini Diversity | | 61 | 7 | 89.01 |
| Local Entropy | | 63 | 5 | 91.65 |
| Local Gini Diversity | | 62 | 6 | 90.62 |
| k-NN SMC (Matlab) | | 60 | 8 | 88.72 |
| k-NN (WEKA) | | 62 | 6 | 91.22 |

## 5.8 Uji Coba Data *Vote*

Dataset *Vote* adalah dataset yang jumlah datanya tergolong sedikit. Namun data ini hanya memiliki 14 atribut, serta kategori label kelasnya hanya ada dua, yaitu *democrat* dan *republican*. Persebaran kelas pada data *Vote* adalah 267 *democrat* dan 168 *republican*. Data akan diuji coba dengan metode *k-Fold Cross Validation* dengan nilai k adalah 10. Untuk hasil uji coba akurasi rata-rata dapat dilihat pada **Tabel 5.6**.

Pada **Tabel 5.6** dapat dilihat bahwa metode Lokal dengan LE memiliki akurasi paling tinggi dibandingkan dengan metode global atau k-NN tanpa pembobotan atribut. Baris yang diberi warna kuning menandakan metode dengan tingkat akurasi paling tinggi.

**Tabel 5.6** Uji Coba Data *Vote*

| Metode | Jumlah Data Rata-rata | Prediksi Benar Rata-rata | Prediksi Salah Rata-rata | Akurasi Rata-rata (%) |
|---|---|---|---|---|
| Global Entropy | 43 | 40 | 3 | 91.95 |
| Global Gini Diversity | | 40 | 3 | 92.18 |
| Local Entropy | | 40 | 3 | 92.41 |
| Local Gini Diversity | | 40 | 3 | 91.03 |
| k-NN SMC (Matlab) | | 40 | 3 | 91.26 |
| k-NN (WEKA) | | 40 | 3 | 91.95 |

## 5.9 Uji Coba Data *Dhermatology*

Dataset *Dhermatology* adalah dataset yang jumlah datanya tergolong sedikit. Meski jumlah datanya sedikit, dataset ini memiliki jumlah atribut yang cukup banyak yaitu 34 atribut. Namun jumlah label kelas dari dataset ini hanya ada enam label kelas saja. Persebaran kelas pada dataset ini cukup merata berkisar 40-70 data per kelasnya. Hanya terdapat 1 kelas yang mencapai seratus data lebih, dan satu kelas lagi yang hanya berjumlah 20 data. Data akan diuji coba dengan metode *k-Fold Cross Validation* dengan nilai k adalah 10. Untuk hasil uji coba akurasi rata-rata dapat dilihat pada **Tabel 5.7**.

**Tabel 5.7** Uji Coba Data *Dhermatology*

| Metode | Jumlah Data Rata-rata | Prediksi Benar Rata-rata | Prediksi Salah Rata-rata | Akurasi Rata-rata (%) |
|---|---|---|---|---|
| Global Entropy | 37 | 34 | 3 | 92.34 |
| Global Gini Diversity | | 34 | 3 | 91.80 |
| Local Entropy | | 35 | 2 | 95.90 |
| Local Gini Diversity | | 35 | 2 | 95.90 |
| k-NN SMC (Matlab) | | 33 | 4 | 90.43 |
| k-NN (WEKA) | | 35 | 2 | 95.35 |

Pada **Tabel 5.7** metode pembobotan Local baik itu LE maupun LG mendapatkan tingkat akurasi yang sama tingginya, lebih tinggi dari metode global maupun tanpa pembobotan. Baris yang diberi warna kuning menandakan metode dengan tingkat

akurasi paling tinggi. Dapat dilihat bahwa terdapat dua metode yang diberi tanda kuning, menandakan ada dua metode yang memiliki tingkat akurasi yang sama dan tertinggi. Hal ini bukan mustahil mengingat bahwa kedua metode yang memiliki tingkat akurasi sama adalah juga sama-sama berasal dari metode lokal. Tentu saja ada kemungkinan keduanya menghasilkan hasil klasifikasi yang tidak jauh berbeda atau bahkan sama persis.

## 5.10 Evaluasi Uji Coba Data Tes

Pada subbab sebelumnya telah diberikan data uji coba berupa tabel untuk masing-masing dataset. Untuk semua dataset, metode yang diajukan baik itu GE, GG, LE, atau LG, dapat memiliki akurasi yang lebih tinggi atau sama dengan k-NN normal tanpa pembobotan atribut (Matlab).

Untuk data setiap data, dari hasil uji coba yang sudah dilakukan, rata-rata metode lokal yang memiliki tingkat akurasi rata-rata yang tinggi. Jika dibandingkan dengan metode lokal, metode global memiliki tingkat akurasi yang sedikit lebih rendah. Namun. Dari hasil uji coba tersebut, dapat dilihat bahwa metode pembobotan atribut rata-rata memiliki tingkat akurasi lebih baik dibandingkan dengan k-NN tanpa pembobotan atribut.

Metode lokal dapat memiliki tingkat akurasi yang lebih tinggi dibandingkan dengan metode global, dikarenakan sifat dari data uji coba itu sendiri. Dalam klasifikasi, data yang diberikan adalah data sebenarnya dari sebuah eksperimen atau uji coba. Yang mana setiap label kelas sudah memiliki ciri tersendiri, hal ini menyebabkan persebaran data seakan-akan telah memiliki kelompok-kelompok tersendiri. Hal ini menyebabkan metode lokal lebih unggul mengingat metode lokal memberi bobot spesifik untuk tiap label kelas yang berbeda. Hal ini dibuktikan dari beberapa data uji coba.

Seperti pada data *Nursery* dan *Cars*, metode global cenderung memiliki tingkat akurasi yang berselisih jauh dengan metode lokal. Ini karena data *Nursery* dan *Cars* memiliki jumlah data yang cukup banyak namun dengan jumlah label kelas yang

sedikit. Berbeda dengan performa untuk data *Soybeans*, yang memiliki jumlah kelas cukup banyak namun jumlah data yang sangat sedikit, metode global dapat memperkecil selisih ketertinggalannya hingga hanya sekitar 2%. Hal ini dapat memberikan sedikit gambaran tentang bagaimana karakteristik tiap metode baik itu lokal maupun global.

# BAB VI
# KESIMPULAN DAN SARAN

Bab ini membahas mengenai kesimpulan yang dapat diambil dari hasil uji coba yang telah dilakukan sebagai jawaban dari rumusan masalah. Selain itu juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut.

## 6.1 Kesimpulan

Kesimpulan yang diperoleh dari uji coba dan evaluasi adalah sebagai berikut:

1. Metode klasifikasi k-NN dapat lebih dimaksimalkan dengan menggunakan pembobotan atribut baik itu global maupun lokal. Pemberian bobot kepada k-NN dapat melalui beberapa metode perhitungan yaitu *Global Entropy*, *Global Gini Diversity*, *Local Entropy*, dan *Local Gini Diversity*.
2. Berdasarkan uji coba yang telah dilakukan, apabila diamati lebih lanjut, pemilihan metode perhitungan bobot menggunakan metode global atau lokal harus mempertimbangkan bentuk persebaran data. Metode global kemungkinan lebih cocok untuk data berjumlah sedikit namun memiliki label kelas yang cukup banyak sehingga tidak ada data yang seakan-akan telah memiliki kelompok tersendiri seperti data *Soybeans* atau data lain yang memiliki jumlah label kelas lebih banyak, sedangkan metode lokal lebih cocok digunakan pada data yang berjumlah banyak dengan label kelas sedikit sehingga data cenderung berkelompok dengan karakteristik tiap label kelas seperti pada data *Nursery*, *Cars*, dan Gerakan Tangan yang telah diuji cobakan sebelumnya.
3. Performa metode k-NN dengan pembobotan atribut dapat dikatakan lebih baik di dalam hal akurasi dibandingkan dengan metode k-NN tanpa pembobotan atribut. Berdasarkan uji coba yang telah dilakukan terhadap

beberapa dataset sebelumnya, terbukti metode k-NN dengan pembobotan atribut memiliki akurasi yang lebih tinggi dibandingkan dengan metode k-NN normal yang dilakukan tanpa pembobotan atribut.

## 6.2 Saran

Saran yang dapat diberikan dalam implementasi k-NN dengan pembobotan atribut adalah sebagai berikut:

1. Perbaikan dataset sebelum dilakukan *training* maupun *testing* untuk menghilangkan *missing value* atau reduksi dimensi dan menghilangkan *noise* (*Preprocessing*)
2. Pemaksimalan metode klasifikasi dengan menggunakan metode perhitungan jarak yang lainnya.
3. Penggunaan metode klasifikasi yang lain, mengingat kompleksitas dari metode klasifikasi k-NN adalah cukup besar yaitu $O(n^2)$.

# LAMPIRAN

## A. Hasil Uji Coba Data *Nursery*

***Training-Testing* 1**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1294 | 1099 | 195 | 84.93 |
| GG | | 1132 | 162 | 87.48 |
| LE | | 1124 | 170 | 86.86 |
| LG | | 1124 | 170 | 86.86 |
| k-NN Normal (Matlab | | 530 | 764 | 40.96 |

***Training-Testing* 2**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1296 | 919 | 377 | 70.91 |
| GG | | 919 | 377 | 70.91 |
| LE | | 953 | 343 | 73.53 |
| LG | | 1060 | 236 | 81.79 |
| k-NN Normal (Matlab | | 432 | 864 | 33.33 |

***Training-Testing* 3**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1296 | 944 | 352 | 72.84 |
| GG | | 944 | 352 | 72.84 |
| LE | | 981 | 315 | 75.69 |
| LG | | 995 | 301 | 76.77 |
| k-NN Normal (Matlab | | 432 | 864 | 33.33 |

***Training-Testing* 4**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1296 | 1006 | 290 | 77.62 |
| GG | | 1006 | 290 | 77.62 |
| LE | | 1050 | 246 | 81.02 |
| LG | | 968 | 328 | 74.69 |
| k-NN Normal (Matlab | | 432 | 864 | 33.33 |

***Training-Testing* 5**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1298 | 916 | 382 | 70.57 |
| GG | | 916 | 382 | 70.57 |
| LE | | 950 | 348 | 73.19 |
| LG | | 966 | 332 | 74.42 |
| k-NN Normal (Matlab | | 432 | 866 | 33.28 |

***Training-Testing* 6**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1294 | 894 | 400 | 69.09 |
| GG | | 895 | 399 | 69.17 |
| LE | | 1074 | 220 | 83.00 |
| LG | | 1077 | 217 | 83.23 |
| k-NN Normal (Matlab | | 432 | 862 | 33.38 |

***Training-Testing* 7**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1296 | 817 | 479 | 63.04 |
| GG | | 819 | 477 | 63.19 |
| LE | | 911 | 385 | 70.29 |
| LG | | 915 | 381 | 70.60 |
| k-NN Normal (Matlab | | 432 | 864 | 33.33 |

***Training-Testing* 8**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1296 | 921 | 375 | 71.06 |
| GG | | 921 | 375 | 71.06 |
| LE | | 911 | 385 | 70.29 |
| LG | | 902 | 394 | 69.60 |
| k-NN Normal (Matlab | | 459 | 837 | 35.42 |

***Training-Testing* 9**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1296 | 994 | 302 | 76.70 |
| GG | | 998 | 298 | 77.01 |
| LE | | 1052 | 244 | 81.17 |
| LG | | 1052 | 244 | 81.17 |
| k-NN Normal (Matlab | | 752 | 544 | 58.02 |

***Training-Testing* 10**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 1298 | 883 | 415 | 68.03 |
| GG | | 882 | 416 | 67.95 |
| LE | | 968 | 330 | 74.58 |
| LG | | 963 | 335 | 74.19 |
| k-NN Normal (Matlab | | 858 | 440 | 66.10 |

## B. Hasil Uji Coba Data *Cars*

***Training-Testing* 1**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 171 | 133 | 38 | 77.78 |
| GG | | 133 | 38 | 77.78 |
| LE | | 141 | 30 | 82.46 |
| LG | | 141 | 30 | 82.46 |
| k-NN Normal (Matlab | | 107 | 64 | 62.57 |

***Training-Testing* 2**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 173 | 130 | 43 | 75.14 |
| GG | | 130 | 43 | 75.14 |
| LE | | 113 | 60 | 65.32 |
| LG | | 113 | 60 | 65.32 |
| k-NN Normal (Matlab | | 105 | 68 | 60.69 |

***Training-Testing* 3**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 173 | 113 | 60 | 65.32 |
| GG | | 113 | 60 | 65.32 |
| LE | | 145 | 28 | 83.82 |
| LG | | 143 | 30 | 82.66 |
| k-NN Normal (Matlab | | 84 | 89 | 48.55 |

***Training-Testing* 4**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 173 | 132 | 41 | 76.30 |
| GG | | 132 | 41 | 76.30 |
| LE | | 143 | 30 | 82.66 |
| LG | | 143 | 30 | 82.66 |
| k-NN Normal (Matlab | | 116 | 57 | 67.05 |

***Training-Testing* 5**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 173 | 132 | 41 | 76.30 |
| GG | | 132 | 41 | 76.30 |
| LE | | 152 | 21 | 87.86 |
| LG | | 152 | 21 | 87.86 |
| k-NN Normal (Matlab | | 122 | 51 | 70.52 |

***Training-Testing* 6**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 173 | 128 | 45 | 73.99 |
| GG | | 128 | 45 | 73.99 |
| LE | | 129 | 44 | 74.57 |
| LG | | 129 | 44 | 74.57 |
| k-NN Normal (Matlab | | 117 | 56 | 67.63 |

***Training-Testing* 7**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 172 | 129 | 43 | 75.00 |
| GG | | 129 | 43 | 75.00 |
| LE | | 127 | 45 | 73.84 |
| LG | | 128 | 44 | 74.42 |
| k-NN Normal (Matlab | | 117 | 55 | 68.02 |

***Training-Testing* 8**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 174 | 140 | 34 | 80.46 |
| GG | | 140 | 34 | 80.46 |
| LE | | 147 | 27 | 84.48 |
| LG | | 144 | 30 | 82.76 |
| k-NN Normal (Matlab | | 123 | 51 | 70.69 |

***Training-Testing* 9**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 172 | 142 | 30 | 82.56 |
| GG | | 142 | 30 | 82.56 |
| LE | | 149 | 23 | 86.63 |
| LG | | 149 | 23 | 86.63 |
| k-NN Normal (Matlab | | 115 | 57 | 66.86 |

***Training-Testing* 10**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 174 | 136 | 38 | 78.16 |
| GG | | 136 | 38 | 78.16 |
| LE | | 145 | 29 | 83.33 |
| LG | | 145 | 29 | 83.33 |
| k-NN Normal (Matlab | | 113 | 61 | 64.94 |

## C. Hasil Uji Coba Data Gerakan Tangan

***Training-Testing* 1**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 10 | 10 | 0 | 100.00 |
| GG | | 10 | 0 | 100.00 |
| LE | | 10 | 0 | 100.00 |
| LG | | 10 | 0 | 100.00 |
| k-NN Normal (Matlab | | 9 | 1 | 90.00 |

***Training-Testing* 2**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 20 | 16 | 4 | 80.00 |
| GG | | 16 | 4 | 80.00 |
| LE | | 17 | 3 | 85.00 |
| LG | | 19 | 1 | 95.00 |
| k-NN Normal (Matlab | | 19 | 1 | 95.00 |

***Training-Testing* 3**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 10 | 10 | 0 | 100.00 |
| GG | | 10 | 0 | 100.00 |
| LE | | 10 | 0 | 100.00 |
| LG | | 10 | 0 | 100.00 |
| k-NN Normal (Matlab | | 10 | 0 | 100.00 |

***Training-Testing* 4**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 20 | 17 | 3 | 85.00 |
| GG | | 17 | 3 | 85.00 |
| LE | | 19 | 1 | 95.00 |
| LG | | 20 | 0 | 100.00 |
| k-NN Normal (Matlab | | 18 | 2 | 90.00 |

***Training-Testing* 5**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 10 | 9 | 1 | 90.00 |
| GG | | 9 | 1 | 90.00 |
| LE | | 10 | 0 | 100.00 |
| LG | | 10 | 0 | 100.00 |
| k-NN Normal (Matlab | | 8 | 2 | 80.00 |

***Training-Testing* 6**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 20 | 18 | 2 | 90.00 |
| GG | | 18 | 2 | 90.00 |
| LE | | 19 | 1 | 95.00 |
| LG | | 19 | 1 | 95.00 |
| k-NN Normal (Matlab | | 17 | 3 | 85.00 |

***Training-Testing* 7**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 10 | 9 | 1 | 90.00 |
| GG | | 9 | 1 | 90.00 |
| LE | | 10 | 0 | 100.00 |
| LG | | 10 | 0 | 100.00 |
| k-NN Normal (Matlab | | 10 | 0 | 100.00 |

***Training-Testing* 8**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 20 | 18 | 2 | 90.00 |
| GG | | 18 | 2 | 90.00 |
| LE | | 18 | 2 | 90.00 |
| LG | | 18 | 2 | 90.00 |
| k-NN Normal (Matlab | | 18 | 2 | 90.00 |

***Training-Testing* 9**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 10 | 9 | 1 | 90.00 |
| GG | | 9 | 1 | 90.00 |
| LE | | 8 | 2 | 80.00 |
| LG | | 9 | 1 | 90.00 |
| k-NN Normal (Matlab | | 10 | 0 | 100.00 |

***Training-Testing* 10**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 20 | 18 | 2 | 90.00 |
| GG | | 18 | 2 | 90.00 |
| LE | | 18 | 2 | 90.00 |
| LG | | 20 | 0 | 100.00 |
| k-NN Normal (Matlab | | 18 | 2 | 90.00 |

## D. Hasil Uji Coba Data *Soybeans*

***Training-Testing* 1**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 64 | 55 | 9 | 85.94 |
| GG | | 55 | 9 | 85.94 |
| LE | | 54 | 10 | 84.38 |
| LG | | 53 | 11 | 82.81 |
| k-NN Normal (Matlab | | 50 | 14 | 78.13 |

***Training-Testing* 2**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 68 | 59 | 9 | 86.76 |
| GG | | 58 | 10 | 85.29 |
| LE | | 63 | 5 | 92.65 |
| LG | | 62 | 6 | 92.18 |
| k-NN Normal (Matlab | | 60 | 8 | 88.24 |

***Training-Testing* 3**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 69 | 66 | 3 | 95.65 |
| GG | | 64 | 5 | 92.75 |
| LE | | 65 | 4 | 94.20 |
| LG | | 64 | 5 | 92.75 |
| k-NN Normal (Matlab | | 61 | 8 | 88.41 |

***Training-Testing* 4**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 68 | 63 | 5 | 92.65 |
| GG | | 63 | 5 | 92.65 |
| LE | | 61 | 7 | 89.71 |
| LG | | 61 | 7 | 89.71 |
| k-NN Normal (Matlab | | 62 | 6 | 91.18 |

***Training-Testing* 5**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 71 | 64 | 7 | 90.14 |
| GG | | 63 | 8 | 88.73 |
| LE | | 65 | 6 | 91.55 |
| LG | | 63 | 8 | 88.73 |
| k-NN Normal (Matlab | | 62 | 9 | 87.32 |

***Training-Testing* 6**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 65 | 53 | 12 | 81.54 |
| GG | | 54 | 11 | 83.08 |
| LE | | 58 | 7 | 89.23 |
| LG | | 56 | 9 | 86.15 |
| k-NN Normal (Matlab | | 57 | 8 | 87.69 |

***Training-Testing* 7**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 67 | 53 | 14 | 79.10 |
| GG | | 54 | 13 | 80.60 |
| LE | | 57 | 10 | 85.07 |
| LG | | 55 | 12 | 82.09 |
| k-NN Normal (Matlab | | 53 | 14 | 79.10 |

***Training-Testing* 8**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 70 | 66 | 4 | 94.29 |
| GG | | 66 | 4 | 94.29 |
| LE | | 67 | 3 | 95.71 |
| LG | | 68 | 2 | 97.14 |
| k-NN Normal (Matlab | | 68 | 2 | 97.14 |

***Training-Testing* 9**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 67 | 63 | 4 | 94.03 |
| GG | | 63 | 4 | 94.03 |
| LE | | 63 | 4 | 94.03 |
| LG | | 64 | 3 | 95.52 |
| k-NN Normal (Matlab | | 63 | 4 | 94 |

***Training-Testing* 10**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 74 | 68 | 6 | 91.89 |
| GG | | 68 | 6 | 91.89 |
| LE | | 73 | 1 | 98.65 |
| LG | | 73 | 1 | 98.65 |
| k-NN Normal (Matlab | | 70 | 4 | 94.59 |

## E. Hasil Uji Coba Data *Vote*

***Training-Testing* 1**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 42 | 41 | 1 | 97.62 |
| GG | | 41 | 1 | 97.62 |
| LE | | 40 | 2 | 95.24 |
| LG | | 39 | 3 | 92.86 |
| k-NN Normal (Matlab | | 39 | 3 | 92.86 |

***Training-Testing* 2**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 44 | 41 | 3 | 93.18 |
| GG | | 41 | 3 | 93.18 |
| LE | | 40 | 4 | 90.91 |
| LG | | 41 | 3 | 93.18 |
| k-NN Normal (Matlab | | 41 | 3 | 93.18 |

***Training-Testing* 3**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 44 | 40 | 4 | 90.91 |
| GG | | 40 | 4 | 90.91 |
| LE | | 43 | 1 | 97.73 |
| LG | | 43 | 1 | 97.73 |
| k-NN Normal (Matlab | | 42 | 2 | 95.45 |

***Training-Testing* 4**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 43 | 37 | 6 | 86.05 |
| GG | | 38 | 5 | 88.37 |
| LE | | 39 | 4 | 90.70 |
| LG | | 39 | 4 | 90.70 |
| k-NN Normal (Matlab | | 36 | 7 | 83.72 |

***Training-Testing* 5**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 44 | 44 | 0 | 100.00 |
| GG | | 44 | 0 | 100.00 |
| LE | | 44 | 0 | 100.00 |
| LG | | 44 | 0 | 100.00 |
| k-NN Normal (Matlab | | 42 | 2 | 95.45 |

***Training-Testing* 6**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 43 | 40 | 3 | 93.02 |
| GG | | 40 | 3 | 93.02 |
| LE | | 40 | 3 | 93.02 |
| LG | | 38 | 5 | 88.37 |
| k-NN Normal (Matlab | | 41 | 2 | 95.35 |

***Training-Testing* 7**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 43 | 41 | 2 | 95.35 |
| GG | | 41 | 2 | 95.35 |
| LE | | 40 | 3 | 93.02 |
| LG | | 39 | 4 | 90.70 |
| k-NN Normal (Matlab | | 41 | 2 | 95.35 |

***Training-Testing* 8**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 44 | 40 | 4 | 90.91 |
| GG | | 40 | 4 | 90.91 |
| LE | | 39 | 5 | 88.64 |
| LG | | 39 | 5 | 88.64 |
| k-NN Normal (Matlab | | 39 | 5 | 88.64 |

***Training-Testing* 9**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 44 | 36 | 8 | 81.82 |
| GG | | 36 | 8 | 81.82 |
| LE | | 38 | 6 | 86.36 |
| LG | | 36 | 8 | 81.82 |
| k-NN Normal (Matlab | | 37 | 7 | 84.09 |

***Training-Testing* 10**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 44 | 40 | 4 | 90.91 |
| GG | | 40 | 4 | 90.91 |
| LE | | 39 | 5 | 88.64 |
| LG | | 38 | 6 | 86.36 |
| k-NN Normal (Matlab | | 39 | 5 | 88.64 |

## F. Hasil Uji Coba Data *Dhermatology*

***Training-Testing* 1**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 35 | 31 | 4 | 88.57 |
| GG | | 32 | 3 | 91.43 |
| LE | | 34 | 1 | 97.14 |
| LG | | 34 | 1 | 97.14 |
| k-NN Normal (Matlab | | 31 | 4 | 88.57 |

***Training-Testing* 2**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 36 | 31 | 5 | 86.11 |
| GG | | 31 | 5 | 86.11 |
| LE | | 35 | 1 | 97.22 |
| LG | | 35 | 1 | 97.22 |
| k-NN Normal (Matlab | | 33 | 3 | 91.67 |

***Training-Testing* 3**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 36 | 33 | 3 | 91.67 |
| GG | | 33 | 3 | 91.67 |
| LE | | 34 | 2 | 94.44 |
| LG | | 34 | 2 | 94.44 |
| k-NN Normal (Matlab | | 31 | 5 | 86.11 |

***Training-Testing* 4**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 36 | 36 | 0 | 100.00 |
| GG | | 35 | 1 | 97.22 |
| LE | | 35 | 1 | 97.22 |
| LG | | 35 | 1 | 97.22 |
| k-NN Normal (Matlab | | 34 | 2 | 94.44 |

***Training-Testing* 5**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 39 | 35 | 4 | 89.74 |
| GG | | 33 | 6 | 84.62 |
| LE | | 38 | 1 | 97.44 |
| LG | | 37 | 2 | 94.87 |
| k-NN Normal (Matlab | | 34 | 5 | 87.18 |

***Training-Testing* 6**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 36 | 34 | 2 | 94.44 |
| GG | | 34 | 2 | 94.44 |
| LE | | 33 | 3 | 91.67 |
| LG | | 33 | 3 | 91.67 |
| k-NN Normal (Matlab | | 34 | 2 | 94.44 |

***Training-Testing* 7**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 36 | 33 | 3 | 91.67 |
| GG | | 33 | 3 | 91.67 |
| LE | | 34 | 2 | 94.44 |
| LG | | 34 | 2 | 94.44 |
| k-NN Normal (Matlab | | 32 | 4 | 88.89 |

***Training-Testing* 8**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 36 | 35 | 1 | 97.22 |
| GG | | 35 | 1 | 97.22 |
| LE | | 36 | 0 | 100.00 |
| LG | | 36 | 0 | 100.00 |
| k-NN Normal (Matlab | | 34 | 2 | 94.44 |

***Training-Testing* 9**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 36 | 33 | 3 | 91.67 |
| GG | | 33 | 3 | 91.67 |
| LE | | 36 | 0 | 100.00 |
| LG | | 36 | 0 | 100.00 |
| k-NN Normal (Matlab | | 34 | 2 | 94.44 |

***Training-Testing* 10**

| Metode | Jumlah Data | Prediksi Benar | Prediksi Salah | Akurasi (%) |
|---|---|---|---|---|
| GE | 40 | 37 | 3 | 92.50 |
| GG | | 37 | 3 | 92.50 |
| LE | | 36 | 4 | 90.00 |
| LG | | 37 | 3 | 92.50 |
| k-NN Normal (Matlab | | 34 | 6 | 85.00 |

# DAFTAR PUSTAKA


[1] "*k*-nearest neighbors algorithm," *Wikipedia*. 28-Nov-2016.

[2] "Instance-based learning," *Wikipedia*. 04-Des-2016.

[3] L. Chen dan G. Guo, "Nearest neighbor classification of categorical data by attributes weighting," *Expert Syst. Appl.*, vol. 42, no. 6, hal. 3142–3149, Apr 2015.

[4] "What is the difference between categorical, ordinal and interval variables?" [Daring]. Tersedia pada: http://www.ats.ucla.edu/stat/mult_pkg/whatstat/nominal_ordinal_interval.htm. [Diakses: 14-Des-2016].

[5] Eka Nugyasa, Yahya. *EKSTRAKSI FITUR DINAMIS PADA GERAKAN TANGAN MENGGUNAKAN KINECT 2.0 UNTUK MENGENALI BAHASA ISYARAT INDONESIA*. 1st ed. Surabaya: Institut Teknologi Sepuluh Nopember, 2017. Print.

[6] "Simple matching coefficient," *Wikipedia*. 15-Agu-2016.

[7] P. A. Jaskowiak dan R. J. G. B. Campello, "Comparing Correlation Coefficients as Dissimilarity Measures for Cancer Classification in Gene Expression Data," dipresentasikan pada BRAZILIAN SYMPOSIUM ON BIOINFORMATICS (BSB), 2011.

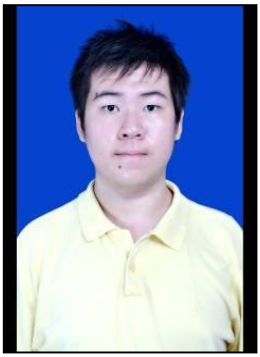
[8] "What is Matlab." [Daring]. Tersedia pada: http://cimss.ssec.wisc.edu/wxwise/class/aos340/spr00/whatismatlab.htm. [Diakses: 14-Des-2016].

[9] "Weka 3 - Data Mining with Open Source Machine Learning Software in Java." [Daring]. Tersedia pada: http://www.cs.waikato.ac.nz/ml/weka/. [Diakses: 14-Des-2016].

[10] "*Cross-validation (statistics)*" *Wikipedia*. 16-Juni-2017.

**[*Halaman ini sengaja dikosongkan*]**

# BIODATA PENULIS



Romario Wijaya, lahir di Surabaya, pada tanggal 12 Desember 1994. Penulis menempuh pendidikan mulai dari SDK Wijana Sejati Mojokerto (2001-2007), SMP Taruna Nusa Harapan Mojokerto (2007-2010), SMA Taruna Nusa Harapan Mojokerto (2010-2013) hingga terakhir Institut Teknologi Sepuluh Nopember Surabaya (2013-2017) di jurusan Teknik Informatika, Fakultas Teknologi Informasi angkatan tahun 2013.

Selama belajar di kampus Teknik Informatika, penulis berkesempatan menjadi asisten dosen Struktur data pada tahun 2016. Selain mengikuti kegiatan akademik, penulis mengikuti kegiatan organisasi sekaligus mengembangkan minatnya pada bidang pengembangan sumber daya manusia dengan bergabung dalam Departemen Pengembangan Sumber Daya Mahasiswa (PSDM) HMTC (2014-2015), Instructor Commitee (2015-2016), penulis juga mengembangan minatnya dalam pemrograman dengan berpartisipasi menjadi panitia National Programming Contest (2014-2015) dan (2015-2016).

Penulis memiliki bidang minat Komputasi Cerdas Visi (KCV) dengan fokus studi pada bidang *image processing*, analisis data, *data mining*, dan *big data*. Komunikasi dengan penulis dapat melalui email: **wijaya.romario@gmail.com**